PREDICTION-POWERED NEURAL ARCHITECTURE SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Evaluating candidate architectures in neural architecture search (NAS) faces an inherent trade-off: on the one hand, reliable performance labels are limited because training and evaluating architectures is costly; on the other hand, zero-cost proxies (ZCPs) are cheap to compute at large scale but can be noisy. Yet, how to effectively combine these two sources of supervision remains unclear. In this paper, we propose PPNAS, a novel prediction-powered inference (PPI) approach for NAS. PPNAS fuses (1) a small set of architectures with observed performance labels and (2) a large set of architectures with ZCP information. To combine these two sources of supervision, PPNAS exploits the ordinal information provided by ZCPs to construct additional pairwise ranking supervision, while PPI debiases systematic discrepancies between ZCP-based and true performance rankings. We evaluate PPNAS in end-to-end predictor-based NAS, where it achieves state-of-the art under limited evaluation budgets. To the best of our knowledge, PPNAS is the first prediction-powered approach for label-efficient NAS.

1 INTRODUCTION

The goal of neural architecture search (NAS) is to identify high-performing neural network architectures. NAS has been successfully applied across a wide range of tasks, including image classification (Zoph & Le, 2016; Real et al., 2019), object detection and semantic segmentation (Poyser & Breckon, 2024), and natural language processing (Elsken et al., 2019; White et al., 2023). However, obtaining reliable performance estimates for candidate architectures typically requires frequent training and evaluation, which is a key bottleneck in NAS, and subject to an inherent trade-off:

(1) *Ground-truth performance labels*, on the one hand, can provide reliable supervision for learning predictors, which can estimate the performance of unseen architectures and guide the search toward promising candidate architectures (White et al., 2020; 2023; Kadlecová et al., 2024). However, training such predictors requires labeled data about architectures with corresponding performance estimates, but obtaining these is costly (Zoph & Le, 2016; Real et al., 2019). As a result, the number of ground-truth labels available for learning accurate performance predictors is limited.

(2) *Zero-cost proxies (ZCPs)*, on the other hand, can provide cheap signals about the performance of candidate architectures *without* the cost of training (Mellor et al., 2021; Abdelfattah et al., 2021; Krishnakumar et al., 2022). To do so, ZCPs assign heuristic scores to candidate architectures based on properties of the neural architecture (e.g., summed importance of specific neurons) or based on other derived quantities computed using a small amount of data (Abdelfattah et al., 2021; Ning et al., 2021; Tanaka et al., 2020; Mellor et al., 2021; Wang et al., 2020; Lee et al., 2018). Prior work suggest that the ordering induced by ZCPs can provide useful ordinal information for ranking different architectures (Dudziak et al., 2020; Abdelfattah et al., 2021; Shen et al., 2023). However, ZCP scores can be noisy and may not reliably reflect true architecture performance.

Together, both ground-truth performance estimates and ZCPs have different strength and weaknesses for evaluating candidate architectures in terms of quantity, computational cost, and reliability. This motivates our research question: ***how can we combine combine supervision signals from (1) reliable but scarce ground-truth performance estimates and (2) scalable but noisy ZCPs?***

To address this challenge, *we propose* PPNAS*, a prediction-powered inference (PPI) approach for label-efficient NAS*. PPNAS fuses (i) a small set of ground-truth labeled architectures with (ii) a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

substantially larger set of ZCP-only samples. We use both sources to construct pairwise ranking supervision, where the ground-truth performance labels provide reliable rankings, while ZCPs provide abundant but noisy rankings at low cost. To combine these two sources of supervision, PPNAS exploits the ordinal information provided by ZCPs, while PPI (Angelopoulos et al., 2023) debiases systematic discrepancies between the ZCP-based rankings and the ground-truth performance rankings. As a result, this allows PPNAS to learn an architecture scoring function that can rank unseen candidate architectures for NAS using only a small number of expensive evaluations. We evaluate PPNAS as a performance-predictor within end-to-end NAS. Our results show that PPNAS improves the performance prediction of candidate architecture which leads to large performance gains in end-to-end NAS under limited budgets.

Methodologically, we adapt principles from PPI (Angelopoulos et al., 2023; 2024) for NAS. PPI provides a principled way to combine large-scale, potentially noisy predictions with scarce ground-truth labels to correct systematic prediction errors and enabling statistically valid inference (Angelopoulos et al., 2023; Zrnic & Candès, 2024). To the best of our knowledge, we are the first to leverage PPI for label-efficient NAS. However, our setting offers three challenges that require careful adaptation. **Challenge** (a): Standard PPI settings focus on estimating a target population parameter. In contrast, in NAS we want to learn a feature-dependent scoring function $g_\theta(X)$ that must generalize to unseen architecture specifications, where the pool of potential architectures is not fixed. This necessitates an adapted prediction-powered objective for ranking and which can then guide architecture search. **Challenge** (b): PPI requires prediction in the same target space, but ZCPs instead provide proxy-specific scores that can only be interpreted through their relative ordering. We thus need to formulate PPI over pairwise ranking losses rather than pointwise prediction losses. **Challenge** (c): The resulting pairwise observations are statistically dependent because multiple pairs can share the same architecture. Consequently, standard variance formulas from PPI++ and power-tuning formulas (Angelopoulos et al., 2024) cannot be applied directly at the pair level; here we instead account for the dependence by computing architecture-level Hájek contributions (Hoeffding, 1948; Serfling, 1980) when controlling the contribution of ZCP supervision and estimation uncertainty.

Our **main contributions** are three-fold:[1] **(1)** We leverage prediction-powered inference for label-efficient NAS. **(2)** We derive a tailored PPI framework for this purpose called PPNAS by developing a rectified pairwise-ranking objective that accounts for systematic discrepancies between ZCP-induced and ground-truth rankings while adaptively weighting the ZCP supervison. **(3)** We demonstrate the practical benefits of PPNAS in end-to-end NAS, where it improves search performance under limited ground-truth evaluation budgets across multiple benchmarks and ZCPs.

2 Related Work

We give an overview of three literature streams relevant to our work: (1) performance prediction in NAS, (2) zero-cost proxies, and (3) prediction-powered inference.

Performance prediction in NAS: Various search strategies have been proposed for NAS, such as reinforcement learning and evolutionary approaches (Zoph & Le, 2016; Real et al., 2019), weight-sharing methods (Bender et al., 2018; Pham et al., 2018; Liu et al., 2018), and predictor-based NAS (White et al., 2020; 2021; Kadlecová et al., 2024). Predictor-based methods, which are the focus of our work, learn a machine learning model to predict architecture performance from the overall architecture specification, and then use these predictions to select promising candidate architectures for subsequent expensive evaluation. Prominent examples of predictor-based methods are **BANANAS** (White et al., 2021), and **BOHAMIANN** (Springenberg et al., 2016), which directly predict validation accuracy. An alternative is learning the relative architecture performance through ranking (Ji et al., 2025). Examples of the latter are **RankNAS** (Hu et al., 2021), **ProxyBO** (Shen et al., 2023), and **ReNAS** (Xu et al., 2021). However, such performance predictors are inherently *supervised* and still rely on costly ground-truth performance evaluations for training; this makes their performance sensitive to the available budget of labeled data. Our approach is orthogonal to predictor-based NAS: PPNAS does not focus on the performance predictor itself but how to leverage the performance predictions for data fusion.

Zero-cost proxies: ZCPs provide noisy but cheap performance estimates, which do *not* require conventional architecture training (Abdelfattah et al., 2021; Mellor et al., 2021; Krishnakumar et al.,

[1] Code for review is available at: `https://anonymous.4open.science/r/ppNAS`.

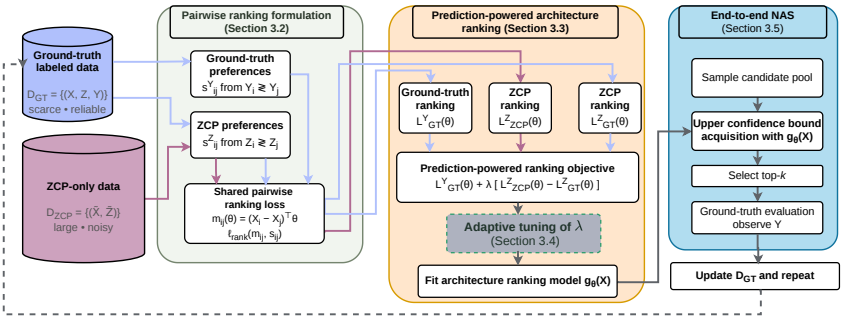


Figure 1: Prediction-powered neural architecture search (PPNAS). We have access to reliable labeled and noisy zero-cost only datasets (left). From both datasets, we construct pairwise ranking supervision (middle-left). The ranking supervision is then used to learn an architecture ranking model with parameters θ, where the contribution of the noisy data is adaptively weighted (middle-right). The trained model is then used in end-to-end NAs to rank candidate architectures (right).

2022). Common examples are `synflow` (Abdelfattah et al., 2021; Tanaka et al., 2020) and `grasp` (Wang et al., 2020). These ZCPs are then used either standalone (Abdelfattah et al., 2021; Mellor et al., 2021; Lin et al., 2021; Li et al., 2023) or as additional features into supervised performance predictors (White et al., 2021; Krishnakumar et al., 2022; White et al., 2023; Kadlecová et al., 2024). Other approaches, such as **ProxyBO** (Shen et al., 2023) or **ZCPBP Warmup** (Abdelfattah et al., 2021), create ZCP-based rankings to augment the training of a predictor. However, existing approaches from this stream primarily use ZCPs as standalone signals, as additional predictor inputs, or directly within the search procedure. In contrast, a principled way to combine the ZCPs with ground-truth architecture evaluations is missing. Our approach is thus different by using ZCP-induced rankings as *additional* supervision for ground-truth rankings.

Prediction-powered inference: PPI provides a principled framework for rigorous statistical inference by "fusing" a small amount of ground-truth labeled data with a much larger set of potentially noisy predictions to improve statistical efficiency while using the ground-truth labels to correct for errors in the predictions (Angelopoulos et al., 2023); **a brief overview is in Section A)**. Several extensions have been developed over the past years. For example, PPI++ extends this approach by a weighting parameter λ to improve the statistical efficiency of the resulting estimator (Angelopoulos et al., 2024). Even other works have adapted PPI to specific tasks such as LLM leaderboards Chatzi et al. (2024); Boyeau et al. (2025); Frauen et al. (2026). However, existing PPI approaches differ from our setting in several important ways: **(1)** In terms of task, PPI has been applied to a range of *inference* problems, but **not** to label-efficient NAS. **(2)** In terms of estimand, rather than ranking a fixed set of observed items, we must learn a feature-dependent scoring function $g_\theta(X)$ that generalizes to previously unseen architectures ($\rightarrow$ Challenge (a)). **(3)** In terms of method, this requires adapting PPI to the dependence structure induced by pairwise architecture comparisons ($\rightarrow$ Challenge (c)).

Research gap: To the best of our knowledge, a principled framework for combining (a) scarce ground-truth architecture evaluations with (b) large-scale ZCPs is missing. Here, we thus introduce the first PPI-based approach to label-efficient NAS.

3 Prediction-powered neural architecture search

3.1 Problem Setting

NAS setup: We consider a standard NAS setting (Kadlecová et al., 2024; Shen et al., 2023; Hu et al., 2021), where the search space is given by $\mathcal{A}$. Each architecture is represented by features $X \in \mathcal{X}$ and has a vector of ZCP scores $Z \in \mathbb{R}^K$ (e.g., the gradient norm (Abdelfattah et al., 2021)). In addition, each architecture has a ground-truth performance $Y \in \mathbb{R}$, such as validation accuracy after training and evaluating the architecture, but which is costly to obtain.

Input: We have access to two datasets: **(a)** a *small labeled* dataset $\mathcal{D}_{\text{GT}} = (X_i, Z_i, Y_i)_{i=1}^{n} \overset{\text{iid}}{\sim} P_{XZY}$, which contains architecture features, ZCP scores, and observed ground-truth (GT) performance; and **(b)** a substantially *larger unlabeled* dataset: $\mathcal{D}_{\text{ZCP}} = (\widetilde{X}_i, \widetilde{Z}_i)_{i=1}^{N} \overset{\text{iid}}{\sim} PXZ$, with $n \ll N$. The latter contains architecture features and ZCP scores only, but no ground-truth performance labels. We

assume that $\mathcal{D}_{\text{GT}}$ and $\mathcal{D}_{\text{ZCP}}$ are independent and that their architecture–proxy pairs are drawn from the same marginal distribution PXZ.

Our goal is to learn an architecture scoring function $g_\theta : \mathcal{X} \to \mathbb{R}$ that ranks unseen candidate architectures according to their ground-truth performance. To learn g_θ, we use both (a) the scarce ground-truth supervision from $\mathcal{D}_{\text{GT}}$ and (b) the large-scale ZCP information from $\mathcal{D}_{\text{ZCP}}$.

Our above task formulation has two important characteristics which will become relevant for our method. *First*, the scoring function g_θ takes architecture features X as input; this allows the learned architecture scoring function to generalize beyond the architectures contained in $\mathcal{D}_{\text{GT}}$ and $\mathcal{D}_{\text{ZCP}}$ and, therefore, to score previously unseen candidate architectures during NAS based on the given features ($\to$ Challenge (a)). *Second*, we note that downstream NAS only requires the relative ordering of architectures; hence, we formulate the learning problem as a ranking task. This also addresses the issue that ZCP scores and ground-truth performance estimates naturally lie on different numerical scales (i.e., ZCPs provide proxy-specific scores, whereas Y may, for example, represent validation accuracy; $\to$ Challenge (b)).

Overview: Our PPNAS has four main components: • **Pairwise ranking formulation**. In Section 3.2, we first define the architecture scoring function and introduce a ranking loss to learn pairwise preferences that is applicable to both ground-truth performance and ZCP scores. • **Prediction-powered architecture ranking**. In Section 3.3, we then combine these two sources of ranking information within a prediction-powered objective. • **Adaptive tuning**. In Section 3.4, we introduce a parameter λ to weight the contribution of the ZCP supervision. • **End-to-end NAS with PPNAS**. In Section 3.5, we describe how we use PPNAS in end-to-end search.

3.2 Pairwise Ranking Formulation

Architecture scoring function. We learn a *architecture scoring function* $g_\theta(X_i) = X_i^\top \theta$ with parameters $\theta \in \mathbb{R}^d$.[2] For a pair of architectures i, j, the predicted ranking margin is $m_{ij}(\theta) = g_\theta(X_i) - g_\theta(X_j) = (X_i - X_j)^\top \theta$. A positive margin indicates that architecture i is predicted to outperform architecture j, while a negative margin indicates the the opposite. Next, we define the pairwise preference targets that encode the desired ordering of an architecture pair based on either ground-truth performance (in $\mathcal{D}_{\text{GT}}$) or ZCP scores (in both $\mathcal{D}_{\text{GT}}$ and $\mathcal{D}_{\text{ZCP}}$). Importantly, expressing both sources through the same pairwise preference representation allows us to formulate a joint ranking loss across the labeled and unlabeled datasets.

Ground-truth pairwise preferences (for $\mathcal{D}_{\text{GT}}$). For two architectures $i, j \in S_n$ with observed ground-truth performances Y_i, Y_j, we define a pairwise preference target via

$$s_{ij}^{Y} = \begin{cases} 1, & Y_i > Y_j, \\ 0, & Y_i < Y_j, \\ \frac{1}{2}, & Y_i = Y_j, \end{cases} \tag{1}$$

where $s_{ij}^{Y} = 1$ indicates that architecture i should outrank architecture j, $s_{ij}^{Y} = 0$ indicates the opposite, and $\frac{1}{2}$ represents a tie.

ZCP-induced pairwise preferences (for $\mathcal{D}_{\text{GT}}$ *and* $\mathcal{D}_{\text{ZCP}}$). Similarly, for a ZCP $Z^{(k)}$, we define ZCP-induced preferences $s_{ij}^{Z,(k)}$ from the ordering of the ZCP scores, analogously to Equation (1):

$$s_{ij}^{Z,(k)} = \begin{cases} 1, & Z_i^{(k)} > Z_j^{(k)}, \\ 0, & Z_i^{(k)} < Z_j^{(k)}, \\ \frac{1}{2}, & Z_i^{(k)} = Z_j^{(k)}, \end{cases} \tag{2}$$

where we assume that larger ZCP scores indicate better performance (this is w.l.o.g. the general interpretation for ZCPs; see (Abdelfattah et al., 2021; Krishnakumar et al., 2022)).

The ranking formulation has a clear benefit: the targets depend *only* on the ordinal information of the ZCP; hence, we do not need to learn a supervised mapping from ZCP scores to validation accuracy. Importantly, both the ground-truth and ZCP-induced preferences are expressed in the same pairwise preference space, which later allows us to adapt PPI ($\to$ Challenge (b)).

[2] Note that an intercept cancels out in the pairwise margin later, and we thus omit it here.

Pairwise ranking loss: To learn the scoring function from pairwise preferences, we employ a pairwise logistic ranking loss

$$\ell_{\text{rank}}(m, s) = -s \log \sigma(m) - (1 - s) \log (1 - \sigma(m)) , \tag{3}$$

where m denotes the predicted ranking margin, $s \in [0, 1]$ the pairwise preference target, and $\sigma(m) = (1 + \exp(-m))^{-1}$ the sigmoid function. Since both ground-truth and ZCP-induced preferences are represented in the same target space, the same ranking loss can be applied to either source of pairwise information.

3.3 Prediction-Powered Architecture Ranking

Rationale for prediction-powered ranking. We now combine the two sources of ranking supervision from $\mathcal{D}_{\text{GT}}$ abd $\mathcal{D}_{\text{ZCP}}$ within a prediction-powered objective. The key challenge in applying PPI to our setting is that ZCPs provide proxy-specific scores, while Y denotes architecture performance estimated on a validation set (i.e., accuracy), hence they are on different scales ($\rightarrow$ Challenge (b)). We therefore apply the PPI principle in the shared space of *pairwise preferences*. To do so, we define three empirical ranking losses: one based on ground-truth preferences constructed the labeled set, and two based on ZCP-induced preferences constructed from the labeled and unlabeled sets, respectively.

Empirical ranking losses. We define three ranking losses corresponding to the supervision available in $\mathcal{D}_{\text{GT}}$ and $\mathcal{D}_{\text{ZCP}}$: **(i)** a ground-truth ranking loss on the labeled set $\mathcal{D}_{\text{GT}}$, **(ii)** a ZCP-induced ranking loss on $\mathcal{D}_{\text{GT}}$, and **(iii)** a ZCP-induced ranking loss on the unlabeled set $\mathcal{D}_{\text{ZCP}}$. For this,let $\mathcal{P}_{\text{GT}} = \{(i, j) : 1 \leq i < j \leq n\}$ and $\mathcal{P}_{\text{ZCP}} = \{(i, j) : 1 \leq i < j \leq N\}$ denote the sets of unique architecture pairs in $\mathcal{D}_{\text{GT}}$ and $\mathcal{D}_{\text{ZCP}}$, respectively.

• **(i)** On the labeled set, we define the *empirical ground-truth ranking loss*:

$$L^{Y}_{\text{GT}}(\theta) = \frac{1}{|\mathcal{P}_{\text{GT}}|} \sum_{(i,j)\in\mathcal{P}_{\text{GT}}} \ell_{\text{rank}} \left(m_{ij}(\theta), s^{Y}_{ij}\right) . \tag{4}$$

This loss measures how well the architecture scoring function agrees with the ordering induced by the observed ground-truth performances. • **(ii)** The *ZCP-induced ranking loss on the labeled set* is

$$L^{Z}_{\text{GT}}(\theta) = \frac{1}{|\mathcal{P}_{\text{GT}}|} \sum_{(i,j)\in\mathcal{P}_{\text{GT}}} \ell_{\text{rank}} \left(m_{ij}(\theta), s^{Z,(k)}_{ij}\right) . \tag{5}$$

which measures agreement with the ZCP-induced preferences available in $\mathcal{D}_{\text{GT}}$. • **(iii)** The *ZCP-induced ranking loss on the unlabeled set* is

$$L^{Z}_{\text{ZCP}}(\theta) = \frac{1}{|\mathcal{P}_{\text{ZCP}}|} \sum_{(i,j)\in\mathcal{P}_{\text{ZCP}}} \ell_{\text{rank}} \left(m_{ij}(\theta), s^{Z,(k)}_{ij}\right) , \tag{6}$$

which analogously measures agreement with the ZCP-induced preferences in $\mathcal{D}_{\text{ZCP}}$.

Prediction-powered ranking objective. We adapt the PPI objective from Section A to our ranking setting by replacing its pointwise losses with the three ranking losses defined above. In addition, following PPI++ (Angelopoulos et al., 2024), we introduce a weighting parameter $\lambda \in [0, 1]$ to control the contribution of the ZCP-induced supervision.

$$\text{The PPNAS loss is thus} \quad \widehat{\theta}_{\lambda} = \arg\min_{\theta} \left\{ L^{Y}_{\text{GT}}(\theta) + \lambda \left[L^{Z}_{\text{ZCP}}(\theta) - L^{Z}_{\text{GT}}(\theta) \right] \right\} . \tag{7}$$

The objective preserves the basic PPI structure while operating entirely on pairwise rankings. The term $L^{Z}_{\text{ZCP}}(\theta)$ incorporates the large-scale ZCP-induced supervision from $\mathcal{D}_{\text{ZCP}}$, whereas $L^{Y}_{\text{GT}}(\theta)$ and $L^{Z}_{\text{GT}}(\theta)$ compare ground-truth and ZCP-induced ranking supervision on the labeled set $\mathcal{D}_{\text{GT}}$. The subtraction of $L^{Z}_{n}(\theta)$ therefore corrects for systematic discrepancies between the ZCP-induced and ground-truth rankings. Importantly, this correction is performed in the *pairwise preference space*, rather than on the numerical ZCP and performance values themselves.

The tuning parameter λ controls the rectifier, that is, how strongly the ZCP-based supervision contributes, and allows us to reduce its influence when the ZCP-induced rankings are less informative. Hence, setting $\lambda = 0$ recovers the ranking objective based on ground-truth supervision, whereas $\lambda = 1$ yields the fully rectified prediction-powered objective. Following PPI++ (Angelopoulos et al., 2024), we adaptively select λ to improve the statistical efficiency; see next section.

3.4 ADAPTIVE TUNING OF λ

Why standard PPI++ cannot be used. Standard PPI++ selects λ to minimize the variance of the estimator (Angelopoulos et al., 2024). However, in our setting, the ranking losses are computed over all architecture pairs, and these pairwise observations are *not* independent, as two pairs can share the same architecture (e.g., (i, j) and (j, k) sharing j; $\rightarrow$ Challenge (c)). Hence, treating the constructed pairs as independent observations would underestimate the variance due to sampling dependence. To address this, we propose to characterize variability at the level of the independently sampled architectures rather than at the level of the constructed pairs.

Architecture-level variance estimates. To obtain architecture-level variance estimates, we estimate the first-order Hájek projection (Hoeffding, 1948; Serfling, 1980) of the gradient of the complete-pair ranking loss. For an architecture pair (i, j), the gradient of the pairwise logistic loss is

$$\psi_{ij}(\theta) = (X_i - X_j)\left[\sigma\left(m_{ij}(\theta)\right) - s_{ij}\right]. \tag{8}$$

For a set B of b architectures with generic empirical ranking loss $U_b(\theta)$, we estimate the architecture-level contribution for architecture $i \in B$ by

$$\widehat{\phi}_i = 2\Big[\frac{1}{b-1}\sum_{j\in B, j\neq i}\psi_{ij}(\theta) - \nabla_\theta U_b(\theta)\Big], \tag{9}$$

where $\psi_{ij}(\theta)$ uses the corresponding ground-truth or ZCP-induced preference target. Intuitively, $\widehat{\phi}_i$ summarizes the first-order contribution of architecture i to the gradient of the complete-pair ranking loss. Mathematically, this yields one contribution per independently sampled architecture, rather than treating the constructed pairs as independent observations. We compute these contributions separately for the ground-truth ranking loss, the ZCP-ranking loss on the labeled set, and the proxy-ranking loss on the unlabeled set, denoted by ϕ_Y, $\phi Z, n$, and $\phi_{Z,N}$, respectively.

Variance-adaptive weighting of ZCP supervision. We next use these architecture-level Hájek contributions to determine how strongly the ZCP-based ranking supervision should enter the rectified objective. On the $\sqrt{n}$ scale, the leading covariance of the rectified estimating equation is

$$\Omega_\lambda = \mathrm{Var}\left(\phi_Y - \lambda\phi_{Z,n}\right) + \frac{n}{N}\lambda^2\,\mathrm{Var}\left(\phi_{Z,N}\right). \tag{10}$$

The first term captures the variability of the rectified ranking gradients on the labeled architectures; hence, increasing λ can reduce this variability when the proxy-induced gradients are informative about the ground-truth gradients. The second term accounts for the additional sampling variability contributed by the ZCP-induced ranking gradients on the unlabeled architectures.

Choice of λ in PPNAS. We select λ by first minimizing the trace of the estimated parameter covariance with fixed $\lambda = 1$:

$$\widehat{\lambda} = \underset{\lambda\in[0,1]}{\arg\min}\ \mathrm{tr}\left(H^\dagger\widehat{\Omega}_\lambda H^{\dagger\top}\right), \tag{11}$$

where $H^\dagger$ denotes the inverse of the Hessian.[3] At this fixed-λ estimate, we compute the

To compute the Hessian, we proceed as follow: We first fit the ranking objective using a fixed value of $\lambda = 1$ and evaluate the Hessian at the resulting parameter estimate. At the same estimate, we compute the architecture-level Hájek contribution covariances and use them to select $\widehat{\lambda}$ according to Equation (11). We then refit the ranking objective using the selected $\widehat{\lambda}$. The procedure thus provides an adaptive variance reduction similar to PPI++ while accounting for the dependence induced by the pairwise ranking construction.

3.5 END-TO-END NAS WITH PPNAS

End-to-end NAS loop. We now build an end-to-end NAS procedure using PPNAS to predict the relative performance of candidate architectures and thus guide the search. The procedure proceeds iteratively over iterations $t = 1, \ldots$ and consists of two steps. **(1) Model update:** At iteration t, we fit PPNAS on the set of architectures evaluated so far, denoted by $\mathcal{D}_{\mathrm{GT}}{}^{(t)}$, and the unlabeled

[3] For numerical stability, we used the pseudoinverse.

architectures, denoted by $\mathcal{D}_{\mathrm{ZCP}}{}^{(t)}$. We then obtain the fitted architecture scoring function $g_{\widehat{\theta}^{(t)}\widehat{\lambda}}$ together with the estimated covariance (i.e., $\widehat{\theta}^{(t)}\widehat{\lambda}$). **(2) Architecture selection:** We use the fitted architecture scoring function to rank the candidate architectures, then select architectures for the next expensive evaluation, and update $\mathcal{D}_{\mathrm{GT}}{}^{(t)}$ with the new observed ground-truth performances to obtain $\mathcal{D}_{\mathrm{GT}}{}^{(t+1)}$. We next describe the acquisition rule used for this selection in step (2).

Uncertainty-aware architecture selection. Inspired by (Xue et al., 2021), we employ an upper confidence bound (UCB) acquisition rule to balance exploitation and exploration. Our acquisition rule is based on the estimated ranking margin and its standard error. We drop t for ease of notation. Let X_b denote the currently best evaluated architecture and X_{cand} a candidate architecture. With $d = X_{\mathrm{cand}} - X_b$, the predicted ranking margin of X_{cand} relative to X_b is $d^\top \hat{\theta}_\lambda$, with estimated standard error $\sqrt{d^\top \widehat{\mathrm{Cov}}(\hat{\theta}_\lambda) d}$. We define the acquisition score as

$$\mathrm{UCB}(X_{\mathrm{cand}}; X_b) = d^\top \hat{\theta}_\lambda + \beta \sqrt{d^\top \widehat{\mathrm{Cov}}(\hat{\theta}_\lambda) d}, \tag{12}$$

where $\beta \geq 0$ controls the exploration–exploitation trade-off (we use $\beta = 0.5$ in our experiments). The first term favors candidates estimated to outperform the current incumbent, whereas the second favors candidates whose relative ranking is estimated less precisely. At iteration t, we evaluate the candidates with the highest acquisition scores and add their observed ground-truth performances to $\mathcal{D}_{\mathrm{GT}}{}^{(t)}$ to obtain $\mathcal{D}_{\mathrm{GT}}{}^{(t+1)}$.

The above procedure has two practical benefits. First, because the acquisition rule operates directly on pairwise ranking margins, it does not require the learned architecture scores to be calibrated to validation accuracy. Second, the covariance estimated by PPNAS directly provides the uncertainty required for acquisition, allowing uncertainty-aware search with a single fitted ranking model rather than an ensemble.

4 Experimental Setup

Benchmark datasets. We use NasBench201 (NB201) (Dong & Yang, 2020), a tabular benchmark that contains pre-computed validation accuracies and ZCPs for neural architectures trained on **CIFAR10** (Krizhevsky, 2009), **CIFAR100** (Krizhevsky, 2009), and **ImageNet16-120** (Chrabaszcz et al., 2017; Dong et al., 2021). To assess whether our findings generalize to other search spaces, we additionally use NASBench101 (NB101) (Ying et al., 2019) and TransNASBench (TNB) (Duan et al., 2021).

Baselines. We compare against the following baselines: ① simple baselines: **ZCP only** (e.g. Krishnakumar et al., 2022; Kadlecová et al., 2024), **random search**; ② non-ranking baselines: **BANANAS** (White et al., 2021), **BOHAMIANN** (Springenberg et al., 2016), **SemiNAS** (Luo et al., 2020); ③ ranking baselines: **ReNAS** (Xu et al., 2021), **PWLNAS** (Ji et al., 2025), **RankNAS** (Hu et al., 2021), **ProxyBO** (Shen et al., 2023), and **ZCPBP Warmup** (Abdelfattah et al., 2021).

ZCPs. We focus on the most-widely used ZCPs from the literature: `synflow` (Abdelfattah et al., 2021; Tanaka et al., 2020), `jacov` and `nwot` (Mellor et al., 2021), `grad_norm` (Abdelfattah et al., 2021), `snip` (Lee et al., 2018), `grasp` (Wang et al., 2020). We encode architectures via a one-hot vector, which is widely used (e.g. White et al., 2020; 2021; Kadlecová et al., 2024).

Eval protocol: We evaluate a label-constrained setting with scarce ground-truth evaluations. Specifically, we use a 30-iteration predictor-guided NAS protocol, starting from $n = 10$ ground-truth labeled architectures and $N = 1000$ ZCP-only architectures. At each iteration, we sample 100 candidates and evaluate the top five according to the respective acquisition rule. We repeat each search 30 times and report mean performance $\pm$ standard deviation. Search quality is measured by final incumbent accuracy and the normalized area under the best-so-far accuracy curve (AUC), which captures how quickly strong architectures are identified. We additionally report mean and median AUC ranks across the evaluated benchmark–dataset–ZCP conditions. Details are in Appendix B.

5 Results

• **Main results.** We first benchmark our proposed PPNAS against the baselines under the same limited ground-truth evaluation budget, so that comparison is *fair*. Results are in Appendix 12, with further search trajectories in Section C. We find: **(1)** Our PPNAS performs best, reaching a 1.5 median rank versus 3 for the second-best method. **(2)** PPNAS consistently reaches a higher AUC

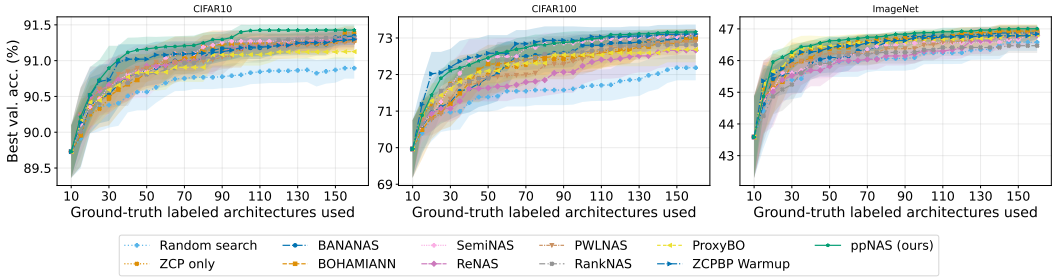


Figure 2: **Main NB201 search results** (with $n = 10$ initial ground-truth-labeled architectures and the `nwot` ZCP). Curves show best-so-far validation accuracy as a function of consumed ground-truth evaluations. PPNAS reaches strong incumbents early in the search, and can reach a higher validation accuracy several iterations earlier.

Table 1: **Aggregated NB201 search results** (with $n = 10$). We report mean final incumbent accuracy and normalized area under the incumbent-accuracy curve (AUC), together with AUC-based mean and median ranks across matched search runs. Best-per-column in bold. Our proposed PPNAS reaches the highest AUC and lowest (i.e., best) ranks.

	CIFAR10		CIFAR100		ImageNet		Rank ↓	
Method	Final Acc	AUC	Final Acc	AUC	Final Acc.	AUC	Mean	Median
Random search	90.89 ± 0.20	90.68 ± 0.28	72.19 ± 0.46	71.52 ± 0.53	46.58 ± 0.36	46.07 ± 0.72	9.44	10.00
ProxyBO	90.99 ± 0.29	90.80 ± 0.27	72.56 ± 0.56	72.14 ± 0.51	46.52 ± 0.43	46.15 ± 0.49	7.83	8.50
ZCP only	90.94 ± 0.40	90.73 ± 0.37	72.48 ± 0.60	72.03 ± 0.57	46.61 ± 0.35	46.20 ± 0.42	7.56	9.50
RankNAS	91.32 ± 0.18	90.96 ± 0.21	72.81 ± 0.51	72.30 ± 0.49	46.47 ± 0.30	46.02 ± 0.38	7.22	7.50
ReNAS	91.30 ± 0.15	90.99 ± 0.21	72.66 ± 0.61	71.93 ± 0.47	46.80 ± 0.37	46.07 ± 0.47	7.11	8.00
PWLNAS	91.26 ± 0.19	90.98 ± 0.19	72.89 ± 0.19	72.15 ± 0.31	46.85 ± 0.25	46.26 ± 0.37	5.94	6.00
BANANAS	91.34 ± 0.10	90.97 ± 0.22	73.00 ± 0.27	72.30 ± 0.37	46.87 ± 0.16	46.33 ± 0.29	5.28	5.50
BOHAMIANN	91.28 ± 0.17	90.98 ± 0.14	72.97 ± 0.37	72.23 ± 0.43	$\mathbf{46.93 \pm 0.27}$	46.42 ± 0.26	4.94	5.00
ZCPBP Warmup	91.16 ± 0.30	90.88 ± 0.28	72.70 ± 0.58	72.25 ± 0.57	46.81 ± 0.29	46.40 ± 0.35	4.94	4.00
SemiNAS	91.31 ± 0.16	91.03 ± 0.23	$\mathbf{73.10 \pm 0.20}$	72.52 ± 0.33	46.84 ± 0.29	46.20 ± 0.28	3.89	3.00
ppNAS (ours)	$\mathbf{91.36 \pm 0.19}$	$\mathbf{91.11 \pm 0.18}$	73.06 ± 0.32	$\mathbf{72.57 \pm 0.41}$	46.87 ± 0.27	$\mathbf{46.45 \pm 0.34}$	**1.83**	**1.50**

than baselines, implying that its ZCP-based supervision can help find better architectures early in the search. ⇒ **Takeaway**: *Our* PPNAS *improves label-efficient NAS search.*

What explains the improved performance? PPNAS differs from baselines in several ways. Here, ReNAS and PWLNAS learn from ground-truth ranking using hinge- and logistic-ranking objectives, respectively, but do *not* exploit the ZCP-only sample. Further, ZCPBP Warmup uses ZCP-induced rankings to warmup a predictor, but not afterwards. In contrast to these ranking-based approaches, PPNAS used the ground-truth and ZCP-induced ranking supervsion *together throughout the entire fitting*. Importantly, we do not treat the proxy rankings as ground truth: their contribution is explicitly rectified using the proxy loss on the gold-labeled sample. The strong performance of PPNAS thus demonstrates that abundant ZCP-induced rankings and (scarce) ground-truth rankings are best combined in a principled way.

• **Breakdown by ZCPs.** We report mean AUC together with mean AUC rank for every ZCP and method for CIFAR10 in Figure 3; further results are in Appenidx I. Across all available NB201 ZCP–dataset combinations, PPNAS consistently ranks first or within the three best methods, except for `synflow` on ImageNet (4th rank). ⇒ **Takeaway**: PPNAS *works with all ZCP-induced supervisions.*

• **Sensitivity: different starting budgets.** We repeat our main experiment but evaluate $n \in \{5, 20, 50\}$ initial samples. Results are in Appendix E. We observe: **(1)** PPNAS maintains the best aggregate rank across all settings (with shared first rank with BANANAS for $n = 50$). **(2)** Differences to the other baselines reduce as more initial samples is available. This is expected: with more initial samples, stronger predictors can be learned from the ground-truth supervision alone.

• **Sensitivity: different unlabeled sizes.** We now vary the size of $\mathcal{D}_{\text{ZCP}}$ across $N \in \{50, 100, 250, 500, 1000\,\text{(default)}, 2000, 3000\}$ for PPNAS. Results are in Appendix F. We observe that performance is relatively stable across the budgets. ⇒ **Takeaway**: PPNAS *can exploit* $\mathcal{D}_{\text{ZCP}}$ *across different sample sizes.*

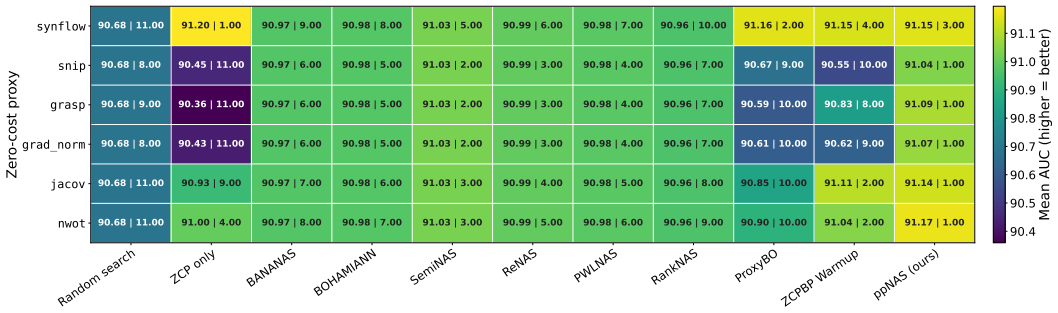


Figure 3: **Breakdown by ZCPs** (on NB201/CIFAR-10). Each cell reports mean AUC | mean AUC rank, color is determined by the AUC. Note that ZCP-independent methods have identical values across rows. PPNAS ranks first except for `synflow`.

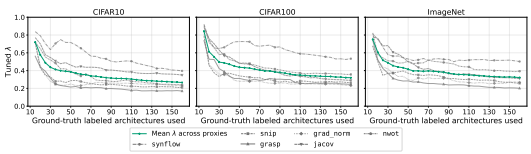


Figure 4: Change of λ over NAS iterations.

• **Analysis of hyperparameter λ.** In Figure 4, we visualize how λ evolves over the search process; further results in AppendixD, Figure 11. We consistently observe that, as more ground-truth evaluation become available, λ decreases from an initial high $\sim$0.8 and stabilizes around 0.2–0.3. Further, the initial λ is higher when fewer ground-truth-labeled architectures are available. This is consistent with the intended role of λ: when ground-truth supervision is scarce, the model places greater weight on the additional ZCP-induced ranking supervision. $\Rightarrow$ **Takeaway**: PPNAS *is effective in adapting λ over the search.*

• **Robustness: different architecture encoding.** We repeat our main experiment and use `arch2vec` representations (Yan et al., 2020) instead. The results are in Appendix G. PPNAS again reaches the best rank. Further, the first four ranks are all held by ranking-based methods, suggesting that `arch2vec` constructs information beneficial for (supervised) ranking. $\Rightarrow$ **Takeaway**: *Our results are robust to different architecture encoding.*

• **Robustness: different search spaces.** We repeat the main experiment on the NB101 and TNB benchmarks. Results are reported in Section H. We find that PPNAS again achieves the best mean and median ranks on both benchmarks. The findings for TNB mean that PPNAS also improves performance for macro-level search spaces. $\Rightarrow$ **Takeaway**: *Our results are robust to different search spaces.*

• **Stress test: misspecified ZCP direction.** Throughout our experiments, we use the common convention that larger ZCP scores indicate better architectures (see Abdelfattah et al. (2021); Krishnakumar et al. (2022) and Figure 17). In practice, a ZCP with the opposite direction could simply be reverse-coded. Here, however, we deliberately leave the direction unchanged to stress-test whether PPNAS can automatically remain robust to such misspecified ZCP supervision. We now use NB101 to stress-test what happens when ZCP direction is misspecified. Here, for `snip`, `grad_norm`, and `jacov`, larger ZCP scores are associated with lower validation performance, as indicated by their negative Spearman correlations in Figure 18.

Despite the ZCP misspecification, PPNAS remains competitive. In the per-ZCP comparison (Figure 19), PPNAS ranks first with `snip` and `jacov`, second with `grad_norm`, and first for all remaining ZCPs. This stress test illustrates that the learned architecture ranking can remain robust even when the direction of the auxiliary ZCP supervision is incorrectly specified. $\Rightarrow$ **Takeaway**: PPNAS *is robust to misspecified ZCP supervision. More broadly, this reflects the role of the prediction-powered formulation, where ZCP-induced preferences provide additional supervision but are not assumed to be perfectly reliable.*

• **Runtimes.** Runtimes for PPNAS are comparable the baselines; see Figure 20.

Conclusion. Our PPNAS is the first prediction-powered approach for label-efficient NAS, by combining scarce ground-truth supervision with large-scale ZCP-induced rankings through a rectified ranking objective. Across our experiments, it achieves consistent performance improvements across datasets, search spaces, and ZCPs. Future work could extend our prediction-powered approach to LLMs and study whether similar ideas can support the study of scaling laws.

STATEMENTS AS PER ICLR POLICY

AI USE STATEMENT

In this work, we used generative AI tools to aid with the implementation of the method described in Section 3. The generated code was checked against reference PPI code available from (Angelopoulos et al., 2023). Further, we used AI tools to create the python code used to create the plots, polish the writing (ensuring fluid transitions between paragraphs, improving the English formulations), and help with literature research. We did not use such tools to formulate mathematical claims, proofs, translation, data generation. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

We feel that our paper submission does not raise any particular concerns w.r.t. the Code of Ethics.

REPRODUCIBILITY STATEMENT

We ensured reproducibility by running all experiments with fixed seeds. Specifically, we used a master seed, from which we *deterministically* derived further seeds used in our code. This includes the seeds used for the data loading/splitting, the random sampling of new candidates during search, the model weight initializations (where applicable), and and intermittent random number generators. Further, our code builds on tabular benchmarks. These provide fixed reference computations of, e.g., validation accuracy and ZCPs. We run our experiments on these benchmarks to ensure a commond ground (as done in other related works). To replicate our PPNAS, we stated the governing math in Section 3, most notably equation 7.

REFERENCES

Mohamed S Abdelfattah, Abhinav Mehrotra, Łukasz Dudziak, and Nicholas D Lane. Zero-cost proxies for lightweight nas. *arXiv preprint arXiv:2101.08134*, 2021.

Anastasios N. Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I. Jordan, and Tijana Zrnic. Prediction-powered inference. *Science*, 382(6671):669–674, 2023. doi: 10.1126/science.adi6000. URL https://www.science.org/doi/abs/10.1126/science.adi6000.

Anastasios N. Angelopoulos, John C. Duchi, and Tijana Zrnic. Ppi++: Efficient prediction-powered inference, 2024. URL https://arxiv.org/abs/2311.01453.

Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *International conference on machine learning*, pp. 550–559. PMLR, 2018.

Pierre Boyeau, Anastasios Nikolas Angelopoulos, Tianle Li, Nir Yosef, Jitendra Malik, and Michael I. Jordan. AutoEval done right: Using synthetic data for model evaluation. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 5276–5290. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/boyeau25a.html.

Ivi Chatzi, Eleni Straitouri, Suhas Thejaswi, and Manuel Gomez Rodriguez. Prediction-powered ranking of large language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 113096–113133. Curran Associates, Inc., 2024. doi: 10.52202/079017-3594. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/cd47cd67caa87f5b1944e00f6781598f-Paper-Conference.pdf.

Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. *arXiv preprint arXiv:2001.00326*, 2020.

Xuanyi Dong, Lu Liu, Katarzyna Musial, and Bogdan Gabrys. Nats-bench: Benchmarking nas algorithms for architecture topology and size. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3634–3646, 2021.

Yawen Duan, Xin Chen, Hang Xu, Zewei Chen, Xiaodan Liang, Tong Zhang, and Zhenguo Li. Transnas-bench-101: Improving transferability and generalizability of cross-task neural architecture search. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5247–5256. IEEE, 2021.

Lukasz Dudziak, Thomas Chau, Mohamed Abdelfattah, Royson Lee, Hyeji Kim, and Nicholas Lane. Brp-nas: Prediction-based nas using gcns. *Advances in neural information processing systems*, 33: 10480–10490, 2020.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: a survey. *J. Mach. Learn. Res.*, 20(1):1997–2017, January 2019. ISSN 1532-4435.

Dennis Frauen, Athiya Deviyani, Mihaela van der Schaar, and Stefan Feuerriegel. Nonparametric LLM evaluation from preference data. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=rHndxbqWyh.

Wassily Hoeffding. A class of statistics with asymptotically normal distribution. *The Annals of Mathematical Statistics*, 19(3):293–325, 1948. ISSN 00034851. URL http://www.jstor.org/stable/2235637.

Chi Hu, Chenglong Wang, Xiangnan Ma, Xia Meng, Yinqiao Li, Tong Xiao, Jingbo Zhu, and Changliang Li. RankNAS: Efficient neural architecture search by pairwise ranking. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 2469–2480, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.191. URL https://aclanthology.org/2021.emnlp-main.191/.

Han Ji, Yuqi Feng, Jiahao Fan, and Yanan Sun. Loss functions for predictor-based neural architecture search. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1624–1633. IEEE, 2025.

Gabriela Kadlecová, Jovita Lukasik, Martin Pilát, Petra Vidnerová, Mahmoud Safari, Roman Neruda, and Frank Hutter. Surprisingly strong performance prediction with neural graph features. *arXiv preprint arXiv:2404.16551*, 2024.

Arjun Krishnakumar, Colin White, Arber Zela, Renbo Tu, Mahmoud Safari, and Frank Hutter. Nas-bench-suite-zero: Accelerating research on zero cost proxies. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 28037–28051. Curran Associates, Inc., 2022. doi: 10.52202/068431-2033. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b3835dd49b7d5bb062aecccc14d8a675-Paper-Datasets_and_Benchmarks.pdf.

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

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

Guihong Li, Yuedong Yang, Kartikeya Bhardwaj, and Radu Marculescu. Zico: Zero-shot nas via inverse coefficient of variation on gradients. *arXiv preprint arXiv:2301.11300*, 2023.

Ming Lin, Pichao Wang, Zhenhong Sun, Hesen Chen, Xiuyu Sun, Qi Qian, Hao Li, and Rong Jin. Zen-nas: A zero-shot nas for high-performance image recognition. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 337–346. IEEE, 2021.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

Renqian Luo, Xu Tan, Rui Wang, Tao Qin, Enhong Chen, and Tie-Yan Liu. Semi-supervised neural architecture search. *Advances in Neural Information Processing Systems*, 33:10547–10557, 2020.

Joe Mellor, Jack Turner, Amos Storkey, and Elliot J Crowley. Neural architecture search without training. In *International conference on machine learning*, pp. 7588–7598. PMLR, 2021.

Xuefei Ning, Changcheng Tang, Wenshuo Li, Zixuan Zhou, Shuang Liang, Huazhong Yang, and Yu Wang. Evaluating efficient performance estimators of neural architectures. *Advances in Neural Information Processing Systems*, 34:12265–12277, 2021.

Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *International conference on machine learning*, pp. 4095–4104. PMLR, 2018.

Matt Poyser and Toby P. Breckon. Neural architecture search: A contemporary literature review for computer vision applications. *Pattern Recogn.*, 147(C), March 2024. ISSN 0031-3203. doi: 10.1016/j.patcog.2023.110052. URL https://doi.org/10.1016/j.patcog.2023.110052.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pp. 4780–4789, 2019.

Robert J. Serfling. *U-Statistics*, chapter 5, pp. 171–209. John Wiley & Sons, Ltd, 1980. ISBN 9780470316481. doi: https://doi.org/10.1002/9780470316481.ch5. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470316481.ch5.

Yu Shen, Yang Li, Jian Zheng, Wentao Zhang, Peng Yao, Jixiang Li, Sen Yang, Ji Liu, and Bin Cui. Proxybo: Accelerating neural architecture search via bayesian optimization with zero-cost proxies. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 9792–9801, 2023.

Jost Tobias Springenberg, Aaron Klein, Stefan Falkner, and Frank Hutter. Bayesian optimization with robust bayesian neural networks. *Advances in neural information processing systems*, 29, 2016.

Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in neural information processing systems*, 33:6377–6389, 2020.

Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*, 2020.

Colin White, Willie Neiswanger, Sam Nolen, and Yash Savani. A study on encodings for neural architecture search. *Advances in neural information processing systems*, 33:20309–20319, 2020.

Colin White, Willie Neiswanger, and Yash Savani. Bananas: Bayesian optimization with neural architectures for neural architecture search. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 10293–10301, 2021.

Colin White, Mahmoud Safari, Rhea Sukthanker, Binxin Ru, Thomas Elsken, Arber Zela, Debadeepta Dey, and Frank Hutter. Neural architecture search: Insights from 1000 papers. *arXiv preprint arXiv:2301.08727*, 2023.

Yixing Xu, Yunhe Wang, Kai Han, Yehui Tang, Shangling Jui, Chunjing Xu, and Chang Xu. Renas: Relativistic evaluation of neural architecture search. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4409–4418. IEEE, 2021.

Song Xue, Bo Zhao, Hanlin Chen, Runqi Wang, and Baochang Zhang. Ucb-enas based on reinforcement learning. In *2021 IEEE 16th Conference on Industrial Electronics and Applications (ICIEA)*, pp. 2008–2013, 2021. doi: 10.1109/ICIEA51954.2021.9516113.

Shen Yan, Yu Zheng, Wei Ao, Xiao Zeng, and Mi Zhang. Does unsupervised architecture representation learning help neural architecture search? *Advances in neural information processing systems*, 33:12486–12498, 2020.

Chris Ying, Aaron Klein, Eric Christiansen, Esteban Real, Kevin Murphy, and Frank Hutter. Nas-bench-101: Towards reproducible neural architecture search. In *International conference on machine learning*, pp. 7105–7114. PMLR, 2019.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

Tijana Zrnic and Emmanuel J. Candès. Cross-prediction-powered inference. *Proceedings of the National Academy of Sciences*, 121(15):e2322083121, 2024. doi: 10.1073/pnas.2322083121. URL https://www.pnas.org/doi/abs/10.1073/pnas.2322083121.

A BACKGROUND ON PPI

We provide a brief overview of PPI (Angelopoulos et al., 2023; 2024). Let $(X, Y) \sim P_{XY}$, where $X \in \mathcal{X}$ denotes the input and $Y \in \mathcal{Y}$ the corresponding ground-truth label. We observe: (a) a small labeled sample $S_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, with $(X_i, Y_i) \overset{\text{iid}}{\sim} P_{XY}$; and (b) a larger, unlabeled sample $\widetilde{S}_N = \{\widetilde{X}_1, \ldots, \widetilde{X}_N\}$, with $n \ll N$ and $\widetilde{X}_i \overset{\text{iid}}{\sim} P_X$, where P_X denotes the marginal distribution of X under P_{XY}. The corresponding ground-truth labels are unobserved for (b). We additionally assume access to a pre-trained predictor $f : \mathcal{X} \to \mathcal{Y}$ to generate noisy, auxiliary labels.

PPI objective. For a loss function ℓ_θ, the goal is to learn a target parameter

$$\theta^* = \arg\min_{\theta \in \Theta} \mathbb{E}\left[\ell_\theta(X, Y)\right]. \tag{13}$$

To do so, we define the empirical ground-truth loss $L_n(\theta)$ and the prediction-based losses $\widetilde{L}^f_N(\theta)$:

$$L_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell_\theta(X_i, Y_i), \quad L^f_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell_\theta(X_i, f(X_i)), \quad \widetilde{L}^f_N(\theta) = \frac{1}{N}\sum_{i=1}^{N} \ell_\theta\Big(\widetilde{X}_i, f(\widetilde{X}_i)\Big). \tag{14}$$

PPI estimates θ^* through the *rectified* prediction-based objective

$$\widehat{\theta}^{\text{PPI}} = \arg\min_{\theta} \left\{ L_n(\theta) + \underbrace{\left[\widetilde{L}^f_N(\theta) - L^f_n(\theta)\right]}_{\text{rectifier}} \right\}. \tag{15}$$

The first term exploits the auxiliary predictions from the large unlabeled sample, while the so-called *rectifier* compares prediction-based and ground-truth losses on the same labeled observations and thereby corrects systematic errors introduced by a noisy predictor f.

B EXPERIMENTAL DETAILS

NAS search protocol. All methods are evaluated within the same iterative predictor-guided NAS protocol. Each search starts from $n = 10$ randomly selected architectures with observed ground-truth validation performance. We additionally provide $N = 1000$ architectures with ZCP information but without ground-truth performance labels. The unlabeled sample size is kept fixed throughout the search.

For each search iteration $t = 1, \ldots, 30$, the corresponding performance predictor is fitted using the information available at that iteration. We then sample 100 candidate architectures from the remaining search space, score them according to the method-specific acquisition rule, and select the top five for ground-truth evaluation. The newly evaluated architectures are added to the labeled set before the next predictor fit. Thus, in the main setting, the labeled set grows from 10 to 160 architectures over the 30 search iterations. For methods that require ensemble-based uncertainty, we use an ensemble of three independently fitted predictors. PPNAS instead uses the parameter covariance estimated by its ranking model for uncertainty-aware acquisition and therefore requires only a single fitted model.

We repeat every search configuration 30 times. At each iteration, we record the validation accuracy of the best architecture observed so far. We summarize each search trajectory using two complementary metrics: final incumbent accuracy at the end of the search and the normalized area under the best-so-far accuracy curve (AUC). We compute AUC over the common 30-iteration search horizon using trapezoidal integration and normalize by the horizon length, such that AUC remains on the same numerical scale as validation accuracy. Final accuracy and AUC are reported as the mean and standard deviation across repeated searches.

To summarize performance across experimental conditions, we rank methods by their AUC within each benchmark–dataset–proxy condition and report the mean and median rank across these comparisons. AUC rewards methods that identify strong architectures early in the search, whereas final accuracy reflects the quality of the incumbent after the complete evaluation budget has been consumed.

Baselines. ProxyBO and ZCPBP Warmup are implementations based on the respective papers, as we could not find publicly usable implementations.

C EXTENDED MAIN RESULTS

We here visualize the search trajectories for other ZCPs. Visualizations are in Figures 5 to 9.

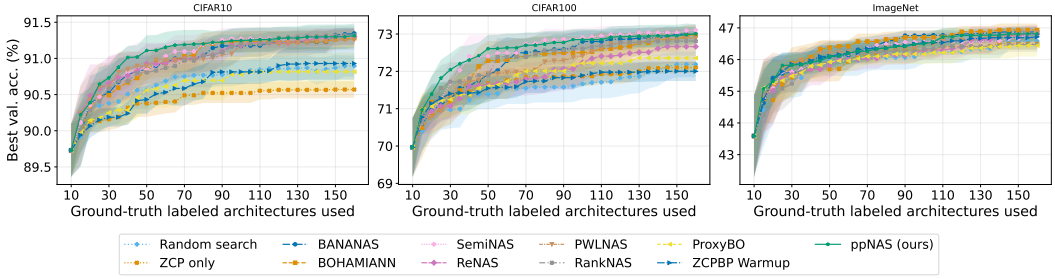


Figure 5: Search trajectory on NB201 using `grad_norm`.

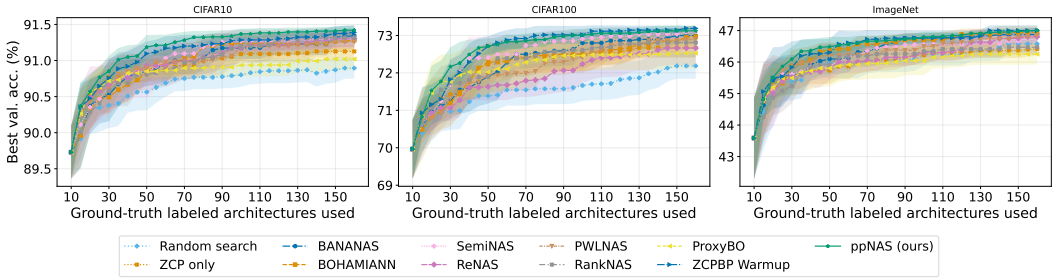


Figure 6: Search trajectory on NB201 using `jacov` (Jacobian covariance).

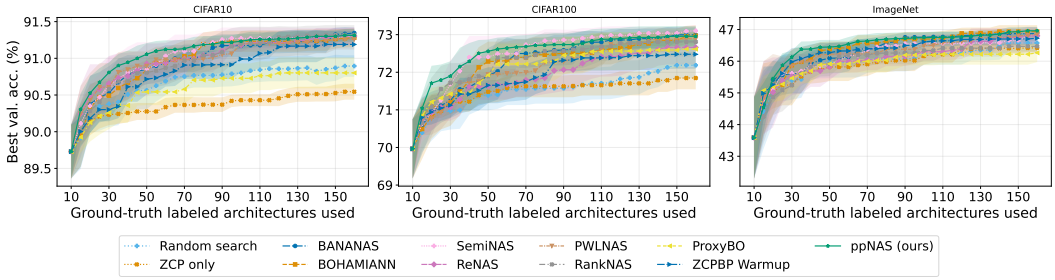


Figure 7: Search trajectory on NB201 using `grasp`.

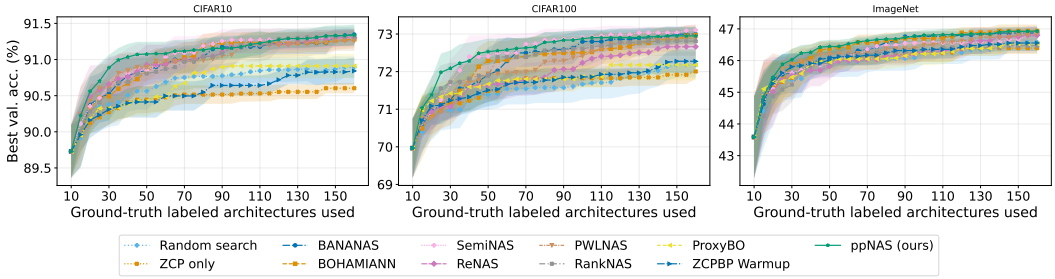


Figure 8: Search trajectory on NB201 using `snip`.

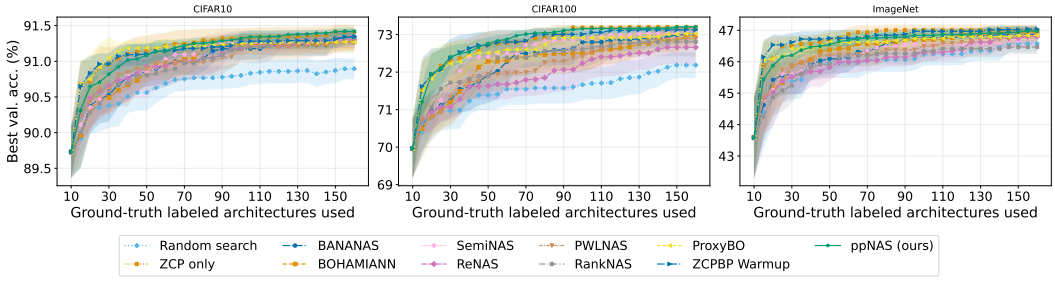


Figure 9: Search trajectory on NB201 using `synflow`.

D λ ANALYSIS

We here provide extended visualizations for the behaviour of the λ weighting parameter over time, see Figure 10. We observe a clear pattern, where the influence of the ZCP rectification is highest in the early iterations, as given by a high λ. As more ground-truth labels become available, the weight slowly decays, until it settles around 0.3. This behavior is consistent for other starting conditions, see Figure 11.

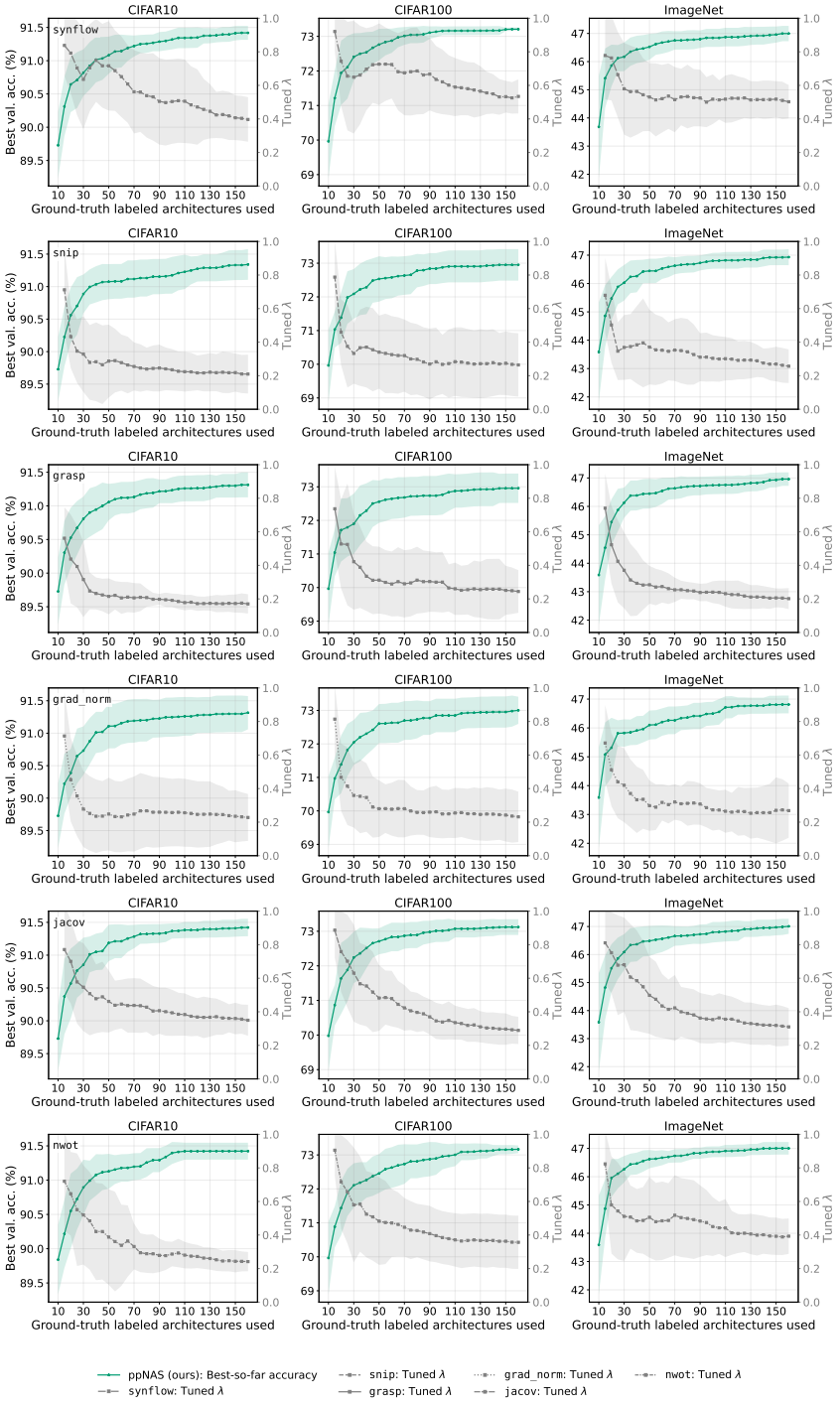
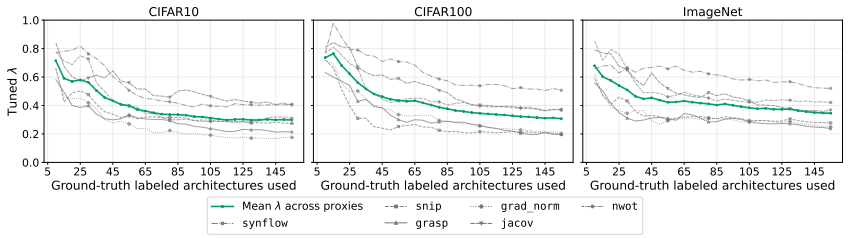
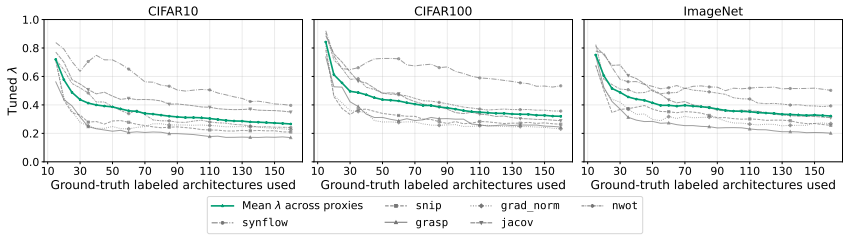


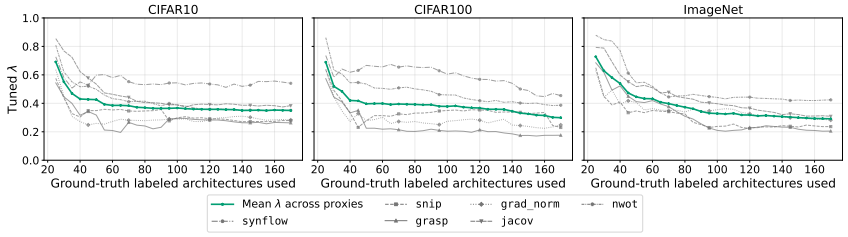
Figure 10: λ as a function of the number of ground-truth labeled architecture acquired during NAS iterations.



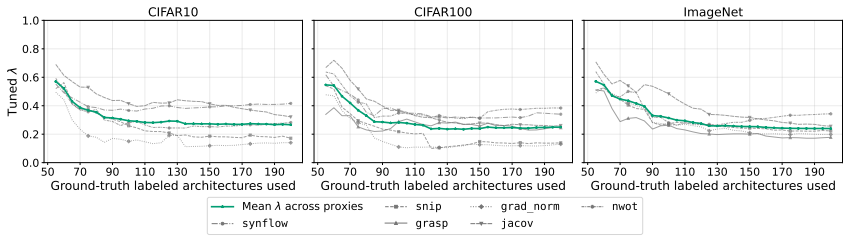
(a) 5 starting points



(b) 10 starting points



(c) 20 starting points



(d) 50 starting points

Figure 11: Evolution of λ over time.

E DIFFERENT INITIAL SIZES

In the main experiments, we use $n = 10$ initial samples to train the performance predictors prior to the first NAS iterations. We here repeat this experiment, again on NB201, but use $n \in \{5, 20, 50\}$. The results are in Tables 2 to 4, with an exemplary search trajectory in Figure 12 for the case of $n = 5$. We observe that at each level of initially labeled samples, our PPNAS outperforms the baselines and reaches the highest final accuracy, AUC, and best ranks.

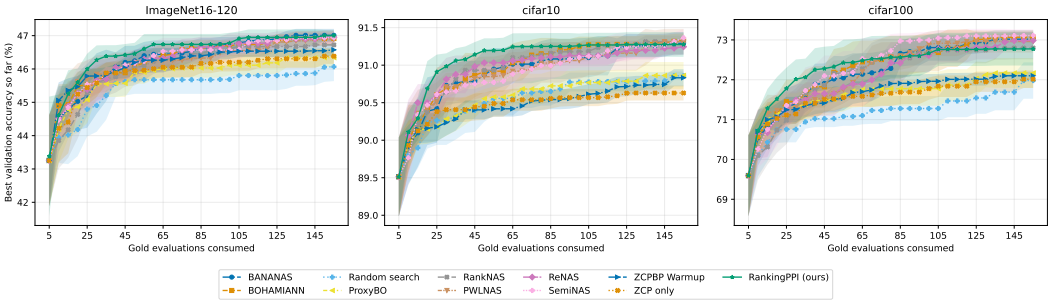


Figure 12: grad norm, gold = 5

Table 2: Ranking across all search experiments on NB201, for 5 initial samples.

	CIFAR10		CIFAR100		ImageNet		Rank ↓	
Method	Final Acc	AUC	Final Acc	AUC	Final Acc.	AUC	Mean	Median
Random search	90.86 ± 0.08	90.55 ± 0.19	71.98 ± 0.61	71.18 ± 0.28	46.07 ± 0.57	45.46 ± 0.51	10.72	11.00
ProxyBO	91.09 ± 0.26	90.81 ± 0.26	72.52 ± 0.62	72.14 ± 0.64	46.54 ± 0.39	46.12 ± 0.42	7.11	8.50
PWLNAS	91.31 ± 0.23	90.91 ± 0.23	72.97 ± 0.38	72.23 ± 0.23	46.89 ± 0.25	46.23 ± 0.31	7.06	7.00
ZCP only	90.93 ± 0.41	90.73 ± 0.35	72.52 ± 0.63	72.09 ± 0.62	46.64 ± 0.35	46.18 ± 0.42	7.00	8.50
ReNAS	91.25 ± 0.14	90.98 ± 0.17	72.99 ± 0.29	72.12 ± 0.32	46.95 ± 0.28	46.28 ± 0.28	6.56	6.50
RankNAS	91.31 ± 0.10	90.99 ± 0.10	72.80 ± 0.38	72.15 ± 0.42	46.72 ± 0.33	46.09 ± 0.37	6.39	7.00
SemiNAS	**91.36 ± 0.16**	90.90 ± 0.19	**73.11 ± 0.21**	72.38 ± 0.30	46.94 ± 0.23	46.28 ± 0.40	5.44	5.00
BANANAS	91.29 ± 0.20	90.92 ± 0.29	73.02 ± 0.29	72.23 ± 0.40	**47.01 ± 0.26**	46.31 ± 0.41	4.94	5.00
ZCPBP Warmup	91.12 ± 0.34	90.87 ± 0.35	72.70 ± 0.56	72.24 ± 0.60	46.82 ± 0.29	46.44 ± 0.35	4.78	4.00
BOHAMIANN	91.33 ± 0.12	90.98 ± 0.23	73.07 ± 0.19	72.30 ± 0.46	46.91 ± 0.29	46.29 ± 0.29	4.11	3.50
ppNAS (ours)	91.33 ± 0.20	**91.10 ± 0.21**	73.00 ± 0.39	**72.45 ± 0.38**	46.93 ± 0.26	**46.46 ± 0.28**	**1.89**	**2.00**

Table 3: Ranking across all search experiments on NB201, for 20 initial samples.

	CIFAR10		CIFAR100		ImageNet		Rank ↓	
Method	Final Acc	AUC	Final Acc	AUC	Final Acc.	AUC	Mean	Median
Random search	90.95 ± 0.16	90.78 ± 0.18	72.01 ± 0.56	71.40 ± 0.34	46.03 ± 0.46	45.71 ± 0.54	10.56	11.00
ZCP only	90.95 ± 0.38	90.78 ± 0.32	72.44 ± 0.63	72.01 ± 0.61	46.66 ± 0.40	46.28 ± 0.39	7.75	9.50
ProxyBO	91.07 ± 0.28	90.85 ± 0.28	72.33 ± 0.66	72.05 ± 0.60	46.58 ± 0.38	46.24 ± 0.44	7.69	9.00
RankNAS	91.27 ± 0.17	91.00 ± 0.22	72.76 ± 0.60	72.12 ± 0.53	46.76 ± 0.25	46.18 ± 0.15	7.25	8.00
ReNAS	**91.38 ± 0.15**	91.00 ± 0.12	72.81 ± 0.42	72.09 ± 0.37	46.66 ± 0.43	46.30 ± 0.39	6.69	6.00
ZCPBP Warmup	91.18 ± 0.32	90.90 ± 0.30	72.77 ± 0.55	72.27 ± 0.58	46.80 ± 0.30	46.42 ± 0.32	5.75	7.00
BOHAMIANN	91.32 ± 0.12	90.96 ± 0.15	72.68 ± 0.47	72.13 ± 0.45	46.92 ± 0.29	46.48 ± 0.24	5.50	6.00
PWLNAS	91.33 ± 0.08	91.08 ± 0.02	72.84 ± 0.35	71.98 ± 0.30	46.85 ± 0.24	46.35 ± 0.30	5.25	5.00
BANANAS	91.31 ± 0.23	91.03 ± 0.21	73.07 ± 0.18	72.37 ± 0.26	46.75 ± 0.20	46.37 ± 0.17	4.56	4.00
SemiNAS	91.33 ± 0.16	91.06 ± 0.10	**73.14 ± 0.09**	72.40 ± 0.29	**46.93 ± 0.33**	46.43 ± 0.34	3.56	3.00
ppNAS (ours)	91.36 ± 0.16	**91.15 ± 0.17**	73.10 ± 0.25	**72.64 ± 0.42**	46.91 ± 0.22	**46.54 ± 0.23**	**1.44**	**1.00**

Table 4: Ranking across all search experiments on NB201, for 50 initial samples.

Method	CIFAR10		CIFAR100		ImageNet		Rank ↓	
	Final Acc	AUC	Final Acc	AUC	Final Acc.	AUC	Mean	Median
Random search	91.10 ± 0.20	90.86 ± 0.16	72.53 ± 0.51	71.99 ± 0.42	46.40 ± 0.43	46.06 ± 0.32	9.94	10.00
ZCP only	90.94 ± 0.36	90.83 ± 0.30	72.39 ± 0.63	72.08 ± 0.57	46.56 ± 0.39	46.30 ± 0.51	9.00	10.00
ProxyBO	91.01 ± 0.27	90.87 ± 0.25	72.44 ± 0.68	72.16 ± 0.58	46.48 ± 0.43	46.26 ± 0.44	9.00	10.00
RankNAS	91.24 ± 0.15	91.07 ± 0.18	72.89 ± 0.37	72.45 ± 0.52	46.67 ± 0.22	46.33 ± 0.29	7.50	8.00
ReNAS	91.26 ± 0.20	91.00 ± 0.16	73.01 ± 0.24	72.50 ± 0.28	46.82 ± 0.41	46.58 ± 0.41	7.25	7.00
ZCPBP Warmup	91.22 ± 0.27	90.99 ± 0.25	72.98 ± 0.39	72.48 ± 0.40	46.88 ± 0.23	46.55 ± 0.31	5.38	6.00
PWLNAS	91.33 ± 0.17	91.15 ± 0.12	72.95 ± 0.33	72.51 ± 0.43	46.89 ± 0.36	46.59 ± 0.36	4.94	5.00
SemiNAS	91.31 ± 0.19	91.06 ± 0.23	72.91 ± 0.54	72.59 ± 0.56	**47.01 ± 0.24**	46.65 ± 0.28	4.75	4.50
BOHAMIANN	91.34 ± 0.13	91.13 ± 0.19	**73.13 ± 0.22**	72.68 ± 0.26	46.96 ± 0.36	46.65 ± 0.45	3.56	3.00
BANANAS	**91.37 ± 0.07**	**91.22 ± 0.13**	73.03 ± 0.28	72.61 ± 0.23	46.99 ± 0.25	**46.68 ± 0.33**	**2.25**	**2.00**
ppNAS (ours)	91.36 ± 0.12	91.18 ± 0.14	73.11 ± 0.25	**72.76 ± 0.31**	46.92 ± 0.26	46.64 ± 0.33	2.44	**2.00**

F DIFFERENT UNLABELED SIZES

In our main experiments, we use $|\mathcal{D}_{\text{ZCP}}| = 1000$. We now vary this for PPNAS to gauge its sensitivity. Results are in Figures 13 and 14. We observe that **(1)** PPNAS can exploit variably-sized $\mathcal{D}_{\text{ZCP}}$, **(2)** search curves show similar patterns across the budgets,i.e. there is no outlier, and **(3)** final performance is only minimally affected.

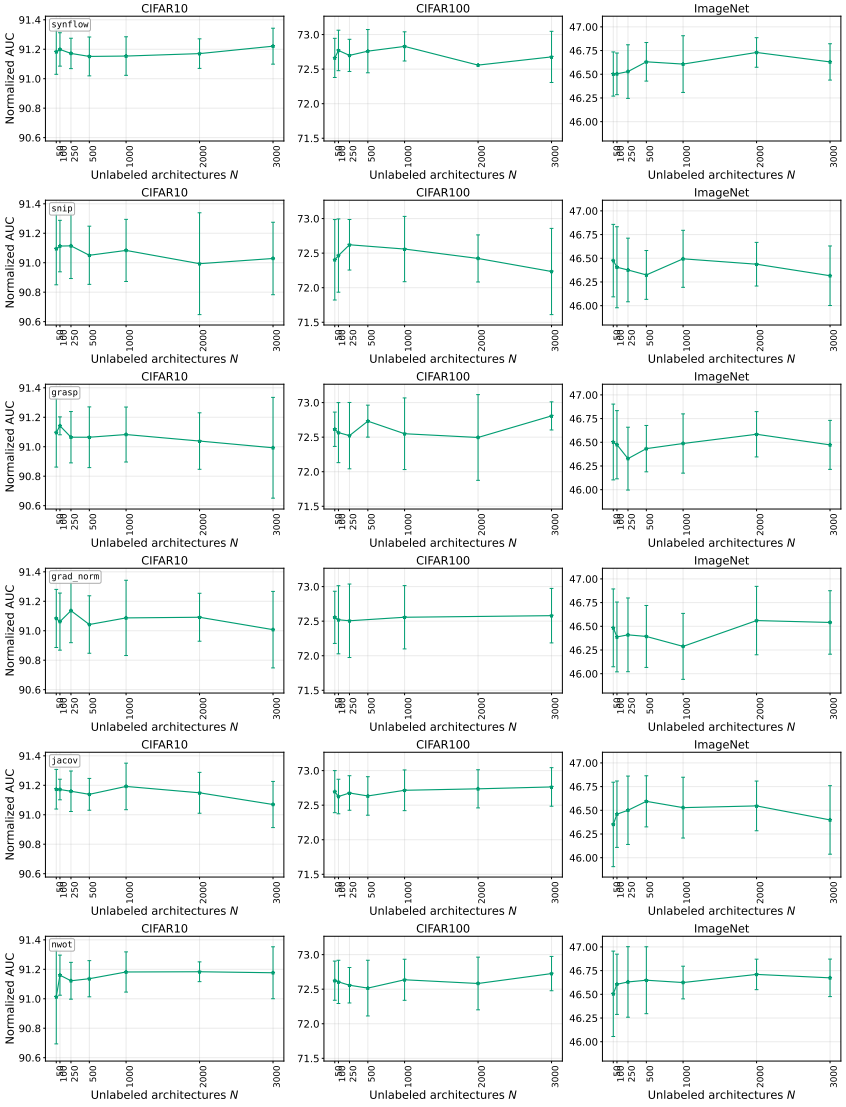


Figure 13: The effect of different N on the final mean AUC.

G OTHER ARCHITECTURE ENCODING

In the main experiments, we use a straightforward one-hot representation. We now repeat this, but use arch2vec (Yan et al., 2020) to encode the neural architecture. Note that we SemiNAS and ProxyBO are not compatbile with this encoding type. Results are in Table 5. PPNAS again performs strongly, reaching the first place by rank.

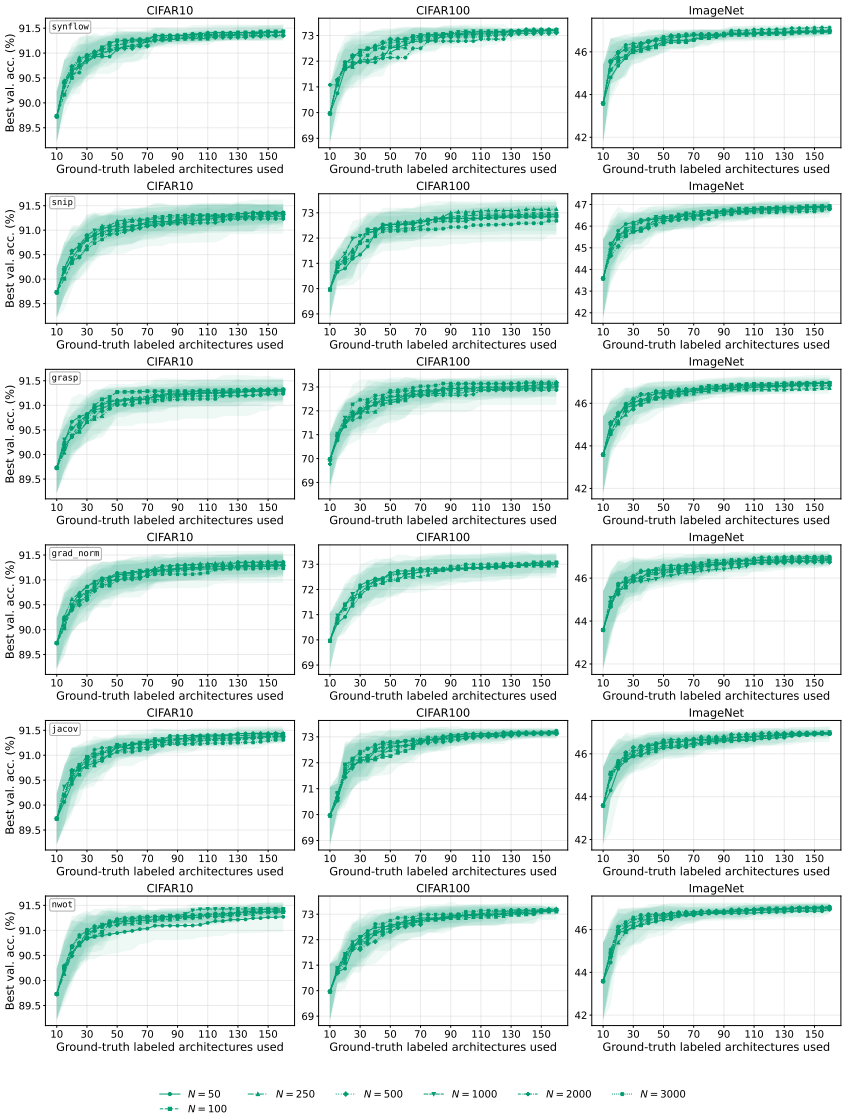


Figure 14: The effect of different N on incumbent curves.

Table 5: Ranking across all search experiments on NB201, *using arch2vec neural architecture representations* (Yan et al., 2020). Our PPNAS reaches the best rank.

	CIFAR10		CIFAR100		ImageNet		Rank ↓	
Method	Final Acc	AUC	Final Acc	AUC	Final Acc.	AUC	Mean	Median
Random search	90.90 ± 0.20	90.68 ± 0.28	72.19 ± 0.46	71.52 ± 0.53	46.58 ± 0.36	46.07 ± 0.72	8.61	9.00
ZCP only	90.94 ± 0.40	90.73 ± 0.37	72.48 ± 0.60	72.03 ± 0.57	46.61 ± 0.35	46.20 ± 0.42	6.89	8.00
BOHAMIANN	91.25 ± 0.14	90.97 ± 0.24	72.88 ± 0.35	72.36 ± 0.42	46.90 ± 0.27	46.44 ± 0.32	6.06	6.00
RankNAS	91.36 ± 0.16	91.06 ± 0.11	72.95 ± 0.26	72.30 ± 0.22	46.84 ± 0.16	46.42 ± 0.18	5.78	6.00
BANANAS	91.34 ± 0.18	91.00 ± 0.12	73.11 ± 0.19	72.46 ± 0.34	46.94 ± 0.32	46.44 ± 0.33	5.06	5.00
ZCPBP Warmup	91.28 ± 0.19	91.00 ± 0.24	73.00 ± 0.36	72.45 ± 0.50	46.79 ± 0.23	46.44 ± 0.32	4.50	5.00
ReNAS	91.40 ± 0.15	91.02 ± 0.20	73.14 ± 0.15	72.60 ± 0.27	46.91 ± 0.22	**46.56 ± 0.24**	3.00	3.00
PWLNAS	**91.43 ± 0.12**	**91.13 ± 0.14**	**73.15 ± 0.14**	72.59 ± 0.36	**47.06 ± 0.24**	46.53 ± 0.31	2.83	3.00
ppNAS (ours)	91.34 ± 0.13	91.12 ± 0.14	73.09 ± 0.25	**72.66 ± 0.34**	46.88 ± 0.26	46.53 ± 0.29	**2.28**	**2.00**

H DIFFERENT SEARCH SPACES

We run our main experiments on the NB201 search spaces. We here repeat the main experiment on the NB101 search space. The results are in Table 6. Our proposed PPNAS again performs best.

Table 6: NB101 CIFAR10 search space.

	CIFAR10		Rank ↓	
Method	Final Acc	AUC	Mean rank	Median rank
Random search	94.17 ± 0.23	94.03 ± 0.30	9.17	9.00
ProxyBO	94.26 ± 0.22	94.06 ± 0.21	8.33	8.00
ZCP only	94.20 ± 0.36	94.01 ± 0.34	7.50	9.00
RankNAS	94.42 ± 0.20	94.18 ± 0.15	6.67	7.00
ZCPBP Warmup	94.47 ± 0.18	94.15 ± 0.23	6.50	5.50
ReNAS	94.42 ± 0.13	94.19 ± 0.09	5.67	6.00
BANANAS	94.55 ± 0.15	94.23 ± 0.13	4.67	5.00
PWLNAS	$\mathbf{94.56 \pm 0.11}$	94.26 ± 0.15	3.17	3.00
BOHAMIANN	94.49 ± 0.17	94.26 ± 0.12	2.17	2.00
ppNAS (ours)	94.54 ± 0.15	$\mathbf{94.31 \pm 0.17}$	**1.17**	**1.00**

Table 7: TNB autoencoder search space.

	AutoEncoder		Rank ↓	
Method	Final Acc	AUC	Mean rank	Median rank
Random search	0.73 ± 0.02	0.70 ± 0.02	9.60	10.00
ReNAS	0.74 ± 0.01	0.71 ± 0.02	8.00	8.00
RankNAS	0.73 ± 0.02	0.71 ± 0.03	7.00	7.00
ZCPBP Warmup	0.73 ± 0.01	0.71 ± 0.02	6.40	6.00
PWLNAS	0.74 ± 0.01	0.71 ± 0.02	6.00	6.00
ZCP only	0.73 ± 0.02	0.71 ± 0.03	5.60	5.00
BANANAS	0.74 ± 0.01	0.72 ± 0.02	4.20	4.00
ProxyBO	0.74 ± 0.01	0.72 ± 0.02	4.00	3.00
BOHAMIANN	0.74 ± 0.01	0.72 ± 0.01	2.60	2.00
ppNAS (ours)	$\mathbf{0.74 \pm 0.01}$	$\mathbf{0.72 \pm 0.01}$	**1.60**	**1.00**

I PER-PROXY RESULTS

We visualize further per-proxy performances in Figures 15 and 16. Our observations are in line with the main results: PPNAS can use a wide range of ZCPs to construct ranking supervision from unsupervised data. It consistently is in the top-3 methods.

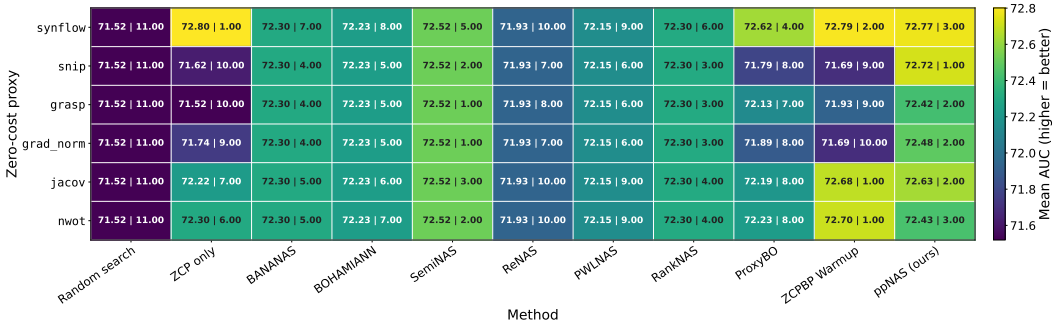


Figure 15: Per-proxy results for CIFAR100 from NB201.

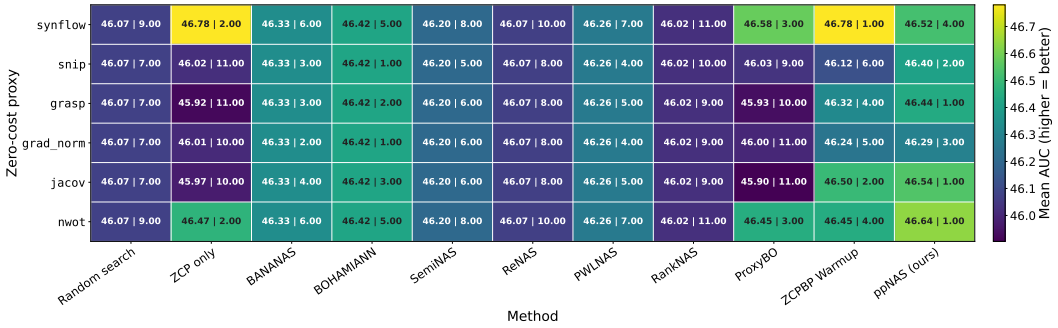


Figure 16: Per-proxy results for ImageNet from NB201.

I.1 STRESS-TEST TO INVERTED PROXY CORRELATIONS

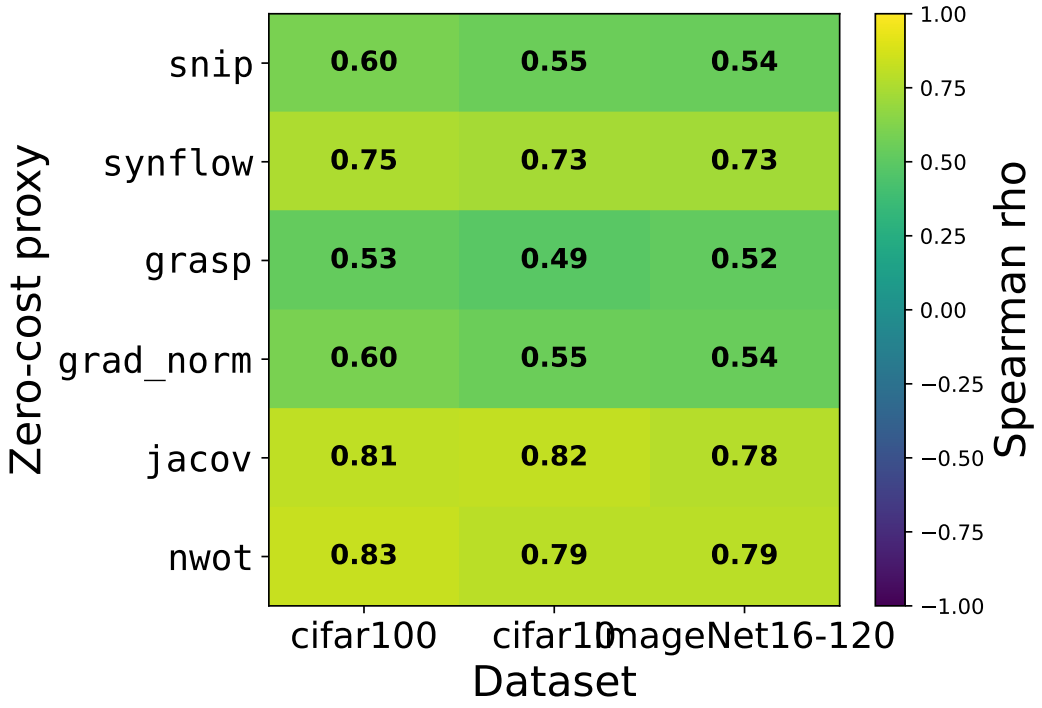


Figure 17: Spearman ranking correlation between zero-cost proxies and validation accuracy for CIFAR10 from the NB101 benchmark.

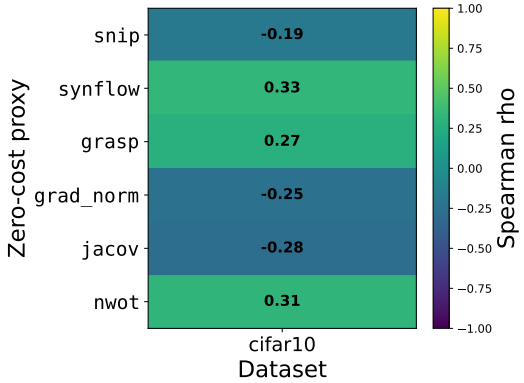


Figure 18: Spearman ranking correlation between zero-cost proxies and validation accuracy for CIFAR10 from the NB101 benchmark.

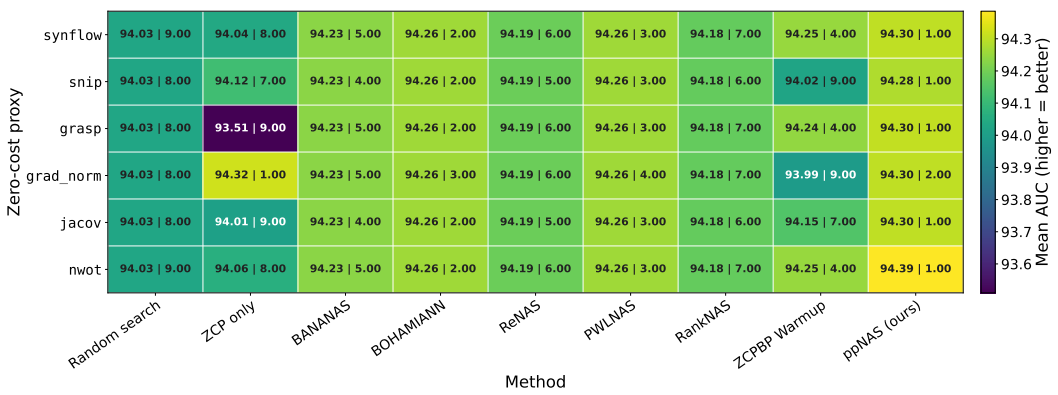


Figure 19: Per-proxy results for CIFAR10 from NB101

J RUNTIME ANALYSIS

We visualize the average runtimes across the NAS iterations in Figure 20. Runtimes of PPNAS are in line with the baselines.

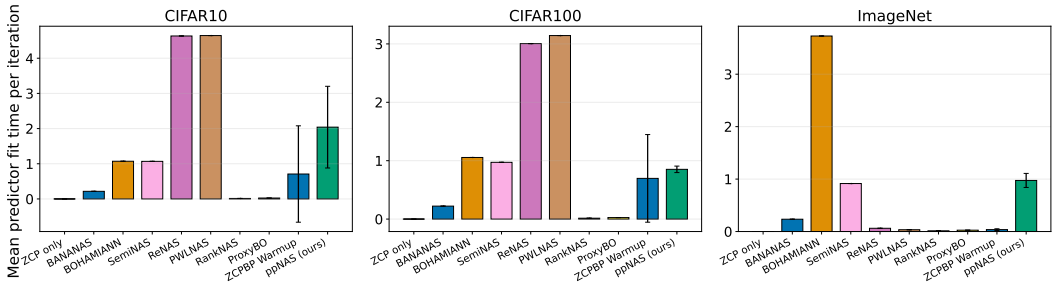


Figure 20: Runtimes. PPNAS is comparable to the baselines.