Master Thesis

Uncertainty Calibration with Online Conformal Prediction in Neural Architecture Search: An Evaluation under the BANANAS Framework

Cheng Chen
(matriculation number 1662473)

July 31, 2025

Submitted to
Data and Web Science Group
Prof. Dr. Margret Keuper
University of Mannheim

Abstract

Some contents

Contents

List of Algorithms

List of Figures

List of Tables

1. Introduction

BANANAS with Conformal Prediction (BANANAS-CP)
Bayesian Optimization with Neural Architectures for Neural Architecture Search (BANANAS)
Neural Architecture Search (NAS)
Conformal Prediction (CP) Split Conformal Prediction (SCP) Conformal Prediction with Cross-validation (CrossVal-CP) Conformal Prediction with Bootstrapping (Bootstrap-CP)

1.1. Motivation

1.2. Related Work

1.3. Contributions and Limitations

1.4. Outline

Having gained an overview of the research question and the background, the remainder of this thesis is organized as follows. First, Chapter 2 reviews the related works on neural architecture search, uncertainty quantification, and in particular, conformal prediction. In Chapter 3, after proposing a novel framework to incorporate uncertainty calibration into the architecture search process in Section 3.1, we describe its methodological steps in more detail. In Section 3.2, we identify different types of conformal prediction algorithms that are applicable for NAS, and consider the use of the underlying surrogate models. In Section 3.3 and Section 3.4, we further examine how the calibrated predictions can be applied in a Bayesian optimization process. In Chapter 4, we describe the benchmark dataset used for conducting experiments and comparing algorithm performance against the state-of-the-art techniques. In Chapter 5, we present the experiment setups and provide interpretations of the results. Finally, Chapter 6 concludes this work and discusses potential future work.

2. Literature Review

2.1. Neural Architecture Search (NAS)

2.1.1. Background

NAS is a subfield of AutoML.

Search Space Details will be introduced in 4

Search Strategy

Performance Evaluation

2.1.2. BANANAS

BANANAS is an Bayesian optimization based search strategy.

Bayesian optimization is a sequential decision-making process that seeks to find a global minimum x argminxX f(x) of an unknown black-box objective function f : X R over an input space X RD.

Give an introduction how Bayesian optimization works. We list the five engineering decisions and review each field's related works. Maybe briefly cite Gaussian Process.

Architecture Encoding

Neural Predictor Feedforward Neural Networks (FNNs)

Uncertainty Estimation

Acquisition Function Independent Thompson Sampling (ITS) Upper Confident Bound (UCB) Probability of Improvements (PI) Expected Improvements (EI)

Acquisition Optimization In each iteration of the Bayesian optimization, the goal is to select a candidate from the search space that maximizes the acquisition score. Evaluating the acquisition function for every architecture available in the search space is computationally infeasible, therefore [14] proposes to create a set of 100 to 1000 candidates and then choose the architecture with the maximal acquisition score in this set. Specifically, [14] explores various approaches for creating this candidate set. The simplest and most natural way is to draw architectures at random. Consider that architectures close in edit distance to those used for training the surrogate model are likely to have more accurate estimates, an alternative is a mutation-based sampling approach, where the candidate set is created via local search by randomly modifying an operation or an edge of the

best-performing architectures that have been evaluated so far. In additional, [14] also examines a hybrid approach that combines random search with mutation-based search. The experiments show that the mutation-based approach outperforms its competitors and suggests it is better to search locally rather than globally.

2.2. Uncertainty Quantification

Understanding uncertainty is important for real-world application of artificial intelligence, e.g., in autonomous driving, medical diagnosis.

2.2.1. Uncertainty Quantification

Aleatory Uncertainty (data uncertainty): uncertainty that arises due to inherent variations and randomness, and cannot be reduced by collecting more information

Epistemic Uncertainty (model uncertainty): uncertainty that arises due to lack of knowledge, and can be reduced by collecting more information.

2.2.2. Uncertainty Estimation Methods

- Bayesian-based: e.g., Bayesian Neural Network
- Ensemble-based: e.g., Monte-Carlo dropout
- Bootstrapping

But these techniques are limited in several perspectives. First, quantifying uncertainty requires training models for several times, which means that the models cannot be applied for real-time prediction or in an online-learning setup. Second, some models are pre-trained and are only accessible via API. Besides, models (pre-)trained on certain datasets may struggle to generalize across different domains or contexts.

2.3. Conformal Prediction

2.3.1. Theoretical Background

Starting from i.i.d data, and provide an intuitive demonstration how the prediction interval is constructed (can add a figure illustrating why conformal prediction works, i.e., symmetry). From the most intuitive expression to the finite-sample adjusted expression.

Notation Then, relax the i.i.d assumption to exchangeability, and lay a formal definition of the conformal prediction. And list the most importance three ingredients of the conformal predictions.

- A trained predictor f
- A conformity score function s. The conformity score is an important engineering decision and has an impact on the size on the prediction set, i.e., the efficiency. The conformity score function can be either a negatively- or positively oriented, in which ... And it can be a random variable as well.

- A target coverage alpha

Marginal coverage is guaranteed regardless of the choices in dataset and black box model. Only the model predictions are required to apply the technique.

A Link to Statistical Testing (clarify the relationship between conformal prediction and hypothesis testing) In this video (22:21), it is explained the intuition why conformal prediction guarantees the coverage, which is quite similar to the spirit of hypothesis testing.

CP	**Hypothesis Testing**
(desired) Coverage level	Confidence level
Nominal error level (1 - Coverage level)	Significance level
The conformity score of the new instance	p-value (is an empirical term)

The coverage parameters which should be pre-set plays a similar role as the confidence interval in hypothesis testing. Conformal prediction is like hypothesis testing with hypotheses:
H0: test instance i conforms to the training instances.
H1: test instance i does not conform to the training instances.

2.3.2. Full Conformal Prediction

Full Conformal Prediction (FCP)

2.3.3. Extensions of Conformal Prediction

Since the transductive version of CP was first proposed in [5], several variants have been developed with different computational complexities, formal guarantees, and practical applications.

To address the aforementioned inefficient computation problem of FCP, Split Conformal Prediction (SCP), also known as Inductive Conformal Prediction (ICP), was first introduced in [10] by replacing the transductive inference with inductive inference. aims to learn a general prediction rule about the data using the observed records. Then, this rule can be applied directly to obtain predictions when new data arrives in sequence, without re-using the training data and retraining the model repeatedly. The main concept involves splitting the data into two non-overlapping subsets, designated for training and calibration, respectively. A predictive model is fit exclusively on the training set, then non-conformity measures are computed on the calibration set to determine the prediction interval's width. Due to its simplicity and computational efficiency, SCP is one

of the most commonly used techniques in the CP family. We delve into methodological steps of SCP with pseudo-code in Section 3.2.1.

———

Limitations of split conformal predictions: - Distribution shift. The conformal prediction is built on the core assumption of exchangeability, which means the data points are identically distributed. However, this assumption is hard to meet in real-world application. For example, with time-series data this assumption is generally violated due to the temporal relationships. - Adaptivity. Once the conformity scores are computed on the calibration set, the decision threshold is settled and is applied to all test datapoints, regardless of the intrinsic complexity of the exact example. It is desirable that the threshold can adapts to the difficulty of the problem and produce a larger prediction interval/set on hard-to-solve example and smaller prediction interval/set on easy-to-solve example. This limitation echoes with the characteristic of Conformal Prediction that the guaranteed coverage is only marginal over all datapoints but not conditional on a specific data points..

Variations of Conformal predictions have been proposed to overcome the limitations. There are three main streams: - find an empirical coverage rate which leads to the desired coverage level. For example, if the desired coverage rate is 90- find an efficient conformity score: Alternatively, [. . .] apply the conformal prediction in an online setting to dynamically incorporate the conformity score of new data points. - find suitable predictor: The trained predictor can be just a poor approximation of the real data generation process.

Besides, [. . .] proposes a CP algorithm that samples datapoints using Monte-Carlo sampling to approach the real distribution of labels in case the ground-truth is ambiguous and consequently cause a biased distribution in manually-annotated labels.

3. Methodology

To address the limitations of the Gaussian assumption in uncertainty estimation, this work introduces a new framework that integrates conformal prediction-based uncertainty calibration into the BANANAS framework in an online setting. An algorithm outlining the overall procedure of BANANAS-CP is presented in Section 3.1, followed by detailed descriptions of each methodological step. Section 3.2 presents different conformal predictions algorithms to be explored. Next in Section 3.3, methods for the estimation and evaluation of the conditional distribution of each candidate architecture are discussed. Finally, in Section 3.4 we introduce how the calibrated distribution can be combined with different acquisition functions and acquisition search strategies in the Bayesian optimization process.

3.1. The BANANAS–CP Framework

Refer to Section 2.1.2 for a detailed introduction of the original BANANAS algorithm. In this section, we emphasis the key ideas of the uncertainty calibration mechanism, as outlined in Step 1 to 6 of the inner iteration in Algorithm 1.

Bayesian optimization is a form of sequential decision-making task. In the applications of neural architecture search, the typical goal is to find the architecture that has the best evaluation performance on a fixed dataset under a given search budget. At each ietration t, a surrogate model is trained on all architectures evaluated at step $\{0, 1, 2..., t-1\}$ and their associated validation accuracies, to predict the scores of unseen architectures for the next search.

In the standard BANANAS setting, the surrogate model is an ensemble of m feed-forward neural networks, typically $m = 5$. At iteration t, a set of candidate architectures is sampled, and a conditional Gaussian distribution is estimated for each candidate based on the ensemble predictions, as expressed below:

$$\hat{f}(a) \sim \mathcal{N}\left(\frac{1}{m}\sum_{i=1}^{m} f_i(a),\ \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(f_i(a) - \frac{1}{m}\sum_{j=1}^{m} f_j(a)\right)^2}\right) \tag{3.1}$$

where a denotes an architecture sampled from the search space, and $f_i(a)$ is the predicted accuracy from the i-th base learner of the ensemble for architecture a.

In the BANANAS-CP framework, a key distinction is that all architectures evaluated at step $\{0, 1, 2..., t-1\}$ are divided disjointly into a training set and a calibration

Algorithm 1 The BANANAS–CP Framework

Input - NAS parameters: search space $\mathcal{A}$, evaluation dataset $\mathcal{D}$, exploration budget T, the number of initially sampled architectures t_0, acquisition function ϕ, surrogate model $\mathcal{M}$ that approximates the true objective function, function $f(\cdot)$ returning validation error of an architecture after training.
Input - Calibration parameters: a function $C(\cdot)$ to create calibration set, a non-conformity score function $s(\cdot)$, and an array of desired quantile levels q.

1: Draw t_0 architectures $\{a_1, a_2, ..., a_{t_0}\}$ uniformly at random from $\mathcal{A}$ and train each individual architecture on $\mathcal{D}$.
2: $\mathcal{A}_{t_0} \leftarrow \{a_1, a_2, ..., a_{t_0}\}$,
3: **for** t in $t_0 + 1, ..., T$ **do**

1. Apply $C(\cdot)$ and split all evaluated architectures into two disjoint datasets; use them as a training set $\mathcal{A}_{t,train}$, and a calibration set $\mathcal{A}_{t,cal}$.
2. Train the surrogate model $\mathcal{M}_t$ on $\{a, f(a)\}, a \in \mathcal{A}_{t,train}$ using the path encoding to represent each architecture.
3. Compute the conformity scores s on $\mathcal{A}_{t,cal}$.
4. Generate a set of candidate architectures from $\mathcal{A}$.
5. **for** each a_i in candidates **do**
   a) Estimate the value for each quantile level q_i in q and calibrate using conformity scores computed in the previous step, with q_i implying a mis-coverage rate $2q_i$ or $2(1 - q_i)$ for conformal prediction.
   b) Fit a distribution F_i based on the estimated quantile values.
   c) Compute the acquisition score $\phi(a_i)$.
6. **end for**
7. Denote a_t as the candidate architecture with maximum $\phi(a)$; evaluate $f(a_t)$.
8. $\mathcal{A}_t \leftarrow \mathcal{A}_{t-1} \cup \{a_t\}$

6: **end for**
7: **Output:** $a^* = \text{argmin}_{t=1,...,T} f(a_t)$

set. Then, the surrogate model is trained exclusively using samples in the training set, while the calibration set is used to compute conformity scores for quantile calibration. In practice, at each iteration t, the surrogate model estimates a conditional distribution $\hat{F}$ for an unseen architecture over its validation accuracy on the target dataset, either based on a specific distribution assumption or a probabilistically-interpretable modeling approach, e.g. quantile regression. Following the definition in [3, 8], calibration means that for any quantile level $p \in [0, 1]$, the empirical fraction of data-points below the p-th percentile of the predicted distribution $\hat{F}$ should converge to p as the sample size goes to infinity. For example, if p = 80%, then the 80th percentile of $\hat{F}$ is set to the threshold value such that 80% of previously evaluated architectures fall below, thereby aligning with the empirical coverage. In an online setting, the objective of the calibration process can be defined as:

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{I}\left\{y_t \leq Q_t(p)\right\} \to p \quad \text{for all } p \in [0,1] \tag{3.2}$$

as $t \to \infty$, where $\mathbb{I}$ is the indicator function and $Q_t(p)$ represents the distribution $\hat{F}$ in the format of quantile function [3, 8].

Next, as in the standard Bayesian optimization process, the acquisition function picks the architecture for the next evaluation based on the conditional distribution of all sampled candidates.

3.2. Uncertainty Calibration Algorithms

As reviewed in Section 2.3, numerous conformal prediction algorithms have been proposed in recent research. This work identifies several approaches applicable in NAS for building a calibration set and computing conformity scores. This section provides an overview of these splitting strategies, as well as the conformity scoring functions that are commonly used for regression problems.

3.2.1. Split Conformal Prediction

To begin, a natural choice for a baseline calibration strategy is the SCP. In this section, we start by introducing the standard SCP procedure, then proceed with the adaptions required to incorporate it into the BANANAS-CP framework.

Implementation steps of SCP are summarized in Algorithm 2. Imagine a regression task where the non-conformity level is measured by the absolute residual, i.e. $|y_i - \hat{y}(x_i)|$. In this case, the algorithm produces a prediction interval for the test point with a width of $[\hat{y}_{test} - \hat{q}\ ,\ \hat{y}_{test} + \hat{q}]$, where $\hat{q}$ is the conformity threshold as defined in line 6.

In this work, we explore SCP in combination with different prediction algorithms. First, we follow the settings in BANANAS and use an ensemble of five FNNs as the underlying surrogate model. In this case, note that the bounds of the prediction set as identified in Algorithm 2 should not be simply interpreted as the quantile values of

Algorithm 2 Split Conformal Prediction

Input: A set of observations $\{(x_i, y_i)\}_{i=1}^{n}$, a prediction algorithm $h(\cdot)$, a non-conformity measure $s(\cdot)$, nominal mis-coverage rate τ, fraction of data assigned to the training set p_{train}, test data x_{n+1}.

Output: a prediction set $\mathcal{C}_\tau(x_{n+1})$ that covers y_{n+1} with probability $1 - \tau$.

1: Allocate at random a proportion of p_{train} of the observations to the training set $\mathcal{D}_{train}$ and use the rest for calibration $\mathcal{D}_{cal}$.
2: Train the point predictor $h(\cdot)$ on $\mathcal{D}_{train}$.
3: Initialise a scoring set $S = \emptyset$
4: **for** (x_i, y_i) in $\mathcal{D}_{cal}$ **do**
$S \leftarrow S \cup \{s(h(x_i), y_i)\}$
5: **end for**
6: Return $\mathcal{C}_\tau(x_{n+1}) \leftarrow \{y \,|\, s((h(x_{n+1}), y) \leq q\}$, where q is the $\lceil (1-\tau)(n_s+1) \rceil$-th smallest value of S, with $n_s = |S|$.

a distribution, since the prediction algorithm does not directly model the τ-quantile of the variable Y, i.e., $Q_Y(\tau) = F_Y^{-1}(\tau) = \inf\{y \colon F_Y(y) \geq \tau\}$, with $\tau \in [0, 1]$ denoting a quantile level and F_Y its cumulative distribution function. Thus, the ensemble predictor must be used in conjunction with a valid distribution assumption to obtain valid quantile values. Motivated by the goal of achieving a completely distribution-agnostic solution, we next replace the ensemble model with a quantile regressor that directly models the quantiles of a distribution. In the remainder of this section, we discuss the configurations designated for each prediction algorithm.

Ensemble Predictor Following the settings in the original BANANAS, an ensemble by default consists of five neural networks, where each neural network is a fully-connected multi-layer perceptron with 20 layers of width 20. The neural networks are trained by minimizing the mean absolute error (MAE), using the Adam optimizer with a learning rate of 0.01. In parallel to BANANAS, we assume that the validation accuracy of each unseen candidate architecture a follows a Gaussian distribution, which is parameterized by the predictive mean ($\hat{\mu}$) and standard deviation ($\hat{\sigma}$) provided by the ensemble model, as demonstrated in equation 3.1. For a specific significance level α (suppose $\alpha < 0.5$), the central quantile interval can be written as:

$$\left[\hat{\mu} - \Phi^{-1}_{1-\alpha/2} \cdot \hat{\sigma} \ , \ \hat{\mu} + \Phi^{-1}_{1-\alpha/2} \cdot \hat{\sigma}\right] \tag{3.3}$$

where $\Phi^{-1}_{1-\alpha/2}$ denotes the $(1 - \frac{\alpha}{2})$-th quantile of the standard normal distribution.

Now, take a closer look at the formula 3.3 and recall the example based on the absolute residuals, which is presented earlier in this section. We observe that the confidence

interval under the Gaussian assumption takes a close form to the prediction interval produced by CP when the conformity scoring function is exactly chosen as:

$$s(\cdot) = \frac{|y_i - \hat{y}(x_i)|}{\hat{\sigma}(x_i)} \tag{3.4}$$

Hence, the bounds of the CP-derived prediction interval can be *approximately* interpreted as empirically calibrated quantile estimates under the Gaussian assumption, provided that the conformity scoring function is chosen appropriately. Note that the absolute residual can be seen as a special case of equation 3.4 as well, where the empirical standard deviation estimate is disregarded and fixed at one. In fact, this scaled absolute residual (equation 3.4) is a popular choice for measuring conformity in practice. Ideally, we would like the CP-derived prediction interval also demonstrates local adaptivity, i.e., the prediction interval should have a larger width if the prediction task is difficult and smaller otherwise. The scaled absolute residual accounts for heteroskedasticity and is able to adjust the width of the prediction band by multiplying the standard deviation estimate. In contrast, the band produced with a pure residual score has constant-width everywhere regardless of the input, which limits its effectiveness in application. Therefore, in this work, we use the scaled absolute residual as the conformity scoring function for ensemble predictors, unless otherwise specified.

Quantile Regressor We now explain how a quantile regressor can be leveraged to build a probabilistic surrogate for Bayesian optimization. We follow the methods established previously in [11, 12].

We start with a brief introduction into the quantile regression [7]. Suppose $(x, y) \sim F$ denote data drawn from a joint distribution that is characterized by its cumulative distribution function F, the aim of the conditional quantile regression is to estimate a given quantile of the conditional distribution of Y given $X = x$. The conditional quantile function for α-quantile is:

$$Q(\alpha) = \inf \{y \in \mathbb{R} : \mathbb{P}(Y \leq y \mid X) \geq \alpha\} \tag{3.5}$$

and can be estimated by minimizing the Pinball loss on the training data [7]:

$$\ell_\alpha(y, \hat{y}) = \begin{cases} \alpha(y - \hat{y}), & \text{if } y \geq \hat{y} \\ (1 - \alpha)(\hat{y} - y), & \text{otherwise} \end{cases} \tag{3.6}$$

where $\hat{y}$ is the predicted quantile value. As illustrated in Figure 3.1, the Pinball loss is asymmetric and the intuition behind is that under-estimate and over-estimate receive different penalties across quantiles. For instance, if $\alpha = 0.9$, then we would expect that empirically 90% of observations should fall below the prediction. In this case, the loss function places a higher penalty for underestimate.

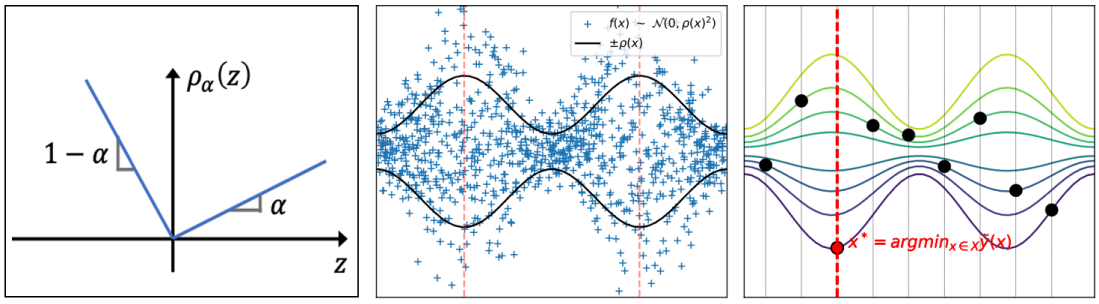

Figure 3.1: in the left is a visualization of the Pinball loss function, where $z = y - \hat{y}$ [11]; the middle displays samples from a synthetic heteroskedasticity function and the right illustrates the sampling procedure based on $|q| = 8$ predicted quantiles [12].

Quantile regression in the BANANAS-CP framework is implemented by training a dedicated neural network for each quantile level q_i in the array q as defined in Algorithm 1 using the corresponding Pinball loss $\ell_{q_i}(y, \hat{y})$.

While quantile regression can model the shape of any continuous distribution given enough data, the predictions are not guaranteed to be well calibrated in practice. In fact, it is not uncommon that quantile regression generates non-monotonic predictions, a phenomenon referred as quantile crossing. To address this issue, we apply a post-hoc calibration upon the predicted quantiles using the Conformal Quantile Regression (CQR) from [11]. This method consists of a novel conformity score tailored for quantile estimation and the key idea of calibration is to apply quantile-aware offsets, which are computed on the calibration set, on the original predicted quantiles.

A close work is [12] that employs CQR to obtain quantiles with robust coverage during hyperparameter tuning via Bayesian optimization. Specifically, the calibrated quantiles are used to select the candidate for the next search, where a set of candidates is first sampled uniformly at random, and then for each of those candidates a random quantile is simply picked and is treated as the acquisition score (Figure 3.1). We follow their notation and interpretation in defining the conformity score for a quantile surrogate:

$$E_i = \max\left\{\hat{q}_{\alpha_j}(x_i) - y_i,\ y_i - \hat{q}_{1-\alpha_j}(x_i)\right\} \tag{3.7}$$

where $\hat{q}_\alpha(x_i)$ denotes the predicted α-quantile at x_i. Note that the sign of the score is positive when the target y_i is outside of the interval and negative when the target falls inside the predicted interval. This allows the conformity score to account for both overcoverage and undercoverage cases. In addition, the score amplitude always measures the distance to the closer quantile between $\hat{q}_{\alpha_j}(x_i)$ and $\hat{q}_{1-\alpha_j}(x_i)$ [11, 12].

3.2.2. Conformal Prediction with Cross-validation

Solving a NAS problem is usually computationally expensive, as each neural architecture evaluation incurs the cost of fully training and validating the underlying model on

the target dataset. Motivated by the fact that NAS based on Bayesian optimization is typically allocated with a budget of 100 to 200 epochs, an additional heuristic for constructing the calibration set via cross-validation (hereafter: CrossVal-CP) is employed to avoid reducing the sample size for obtaining a holdout set as performed in SCP.

The CrossVal-CP method is a natural extension of SCP and is first formally presented in [13]. At each step, the evaluated architectures are devided at random into K folds. A dedicated surrogate model is trained on $K-1$ folds, while the remaining one is used as the calibration set to calculate the conformity scores. This process is repeated for K times over each individual fold. Finally, conformity scores from all calibration folds are combined to form the overall calibration set, on which the quantile is computed to determine the calibration offset. For an unseen data point, the prediction is obtained by aggregating the predictions of the K trained models, e.g., by taking the average of the model outputs. Algorithm 3 summarizes this procedure.

Algorithm 3 Conformal Prediction with Cross-validation

Input: A set of observations $\{(x_i, y_i)\}_{i=1}^{n}$, number of folds K, a prediction algorithm $h(\cdot)$, a non-conformity measure $s(\cdot)$, nominal mis-coverage rate τ, test data x_{n+1}.
Output: a prediction set $\mathcal{C}_\tau(x_{n+1})$ that covers y_{n+1} with probability $1-\tau$.

1: Initialise a conformity scoring set $S = \emptyset$
2: Split the observations $\{(x_i, y_i)\}_{i=1}^{n}$ into K folds at random. I_k denotes the index set containing indices of samples in the k-th fold.
3: **for** k in 1, 2, ..., K **do**

1. Train $\hat{h}_{-k}(\cdot)$ on $\{(x_i, y_i) \mid i \notin I_k\}$
2. Compute conformity score on the k-th fold $S_k = \{s((\hat{h}_{-k}(x_i), y_i) \mid i \in I_k\}$
3. $S \leftarrow S \cup S_k$

4: **end for**
5: Predict x_{n+1}: $h(x_{n+1}) \leftarrow aggregate(\{\hat{h}_{-1}(x_{n+1}), ..., \hat{h}_{-K}(x_{n+1})\})$
6: Return $\mathcal{C}_\tau(x_{n+1}) \leftarrow \{y \,|\, s((h(x_{n+1}), y) \leq q\}$, where q is the $\lceil (1-\tau)(n_s+1) \rceil$-th smallest value of S, with $n_s = |S|$.

Note that the only distinction between SCP and CrossVal-CP is how the calibration set is constructed. Since it does not place any additional restriction on the choices of the underlying surrogate, CrossVal-CP can be applied in conjunction with either ensemble predictor or quantile regressor in the same way as SCP. See Section 3.2.1 for detailed configurations.

3.2.3. Conformal Prediction with Bootstrapping

Inspired by the fact that BANANAS is built on an ensemble surrogate, we further explore incorporating Jackknife+-after-boostrap [6], a wrapper for predictive inference

Algorithm 4 Conformal Prediction with Bootstrapping

Input: A set of observations $\{(x_i, y_i)\}_{i=1}^n$, number of bootstraps B, a prediction algorithm $h(\cdot)$, a non-conformity measure $s(\cdot)$, nominal mis-coverage rate τ, test data x_{n+1}.

Output: a prediction set $\mathcal{C}_\tau(x_{n+1})$ that covers y_{n+1} with probability $1 - \tau$.

1: Sample all available data with replacement and create B subsets. I_b denotes the indices of data points included in the b-th sample.
2: Train $\hat{h}_b(\cdot)$ on $\{(x_i, y_i) \mid i \in I_b\}$ for b in 1, 2, ..., B
3: Initialise a conformity scoring set $S = \emptyset$
4: **for** i in 1, 2, ..., n **do**

1. Initialize an empty for leave-one-out estimates $LOO_i = \emptyset$
2. For b in 1, 2, ..., B, if $i \notin \mathrm{I}_b$: $LOO_i \leftarrow LOO_i \cup \hat{h}_b(x_i)$

3. S $\leftarrow S \cup s\big(aggregate(LOO_i),\ y_i\big)$

8: **end for**
9: Predict x_{n+1}: $h(x_{n+1}) \leftarrow aggregate(\{\hat{h}_1(x_{n+1}), ..., \hat{h}_B(x_{n+1})\})$
10: Return $\mathcal{C}_\tau(x_{n+1}) \leftarrow \{y \,|\, s((h(x_{n+1}), y) \leq q\}$, where q is the $\lceil (1-\tau)(n_s+1) \rceil$-th smallest value of S, with $n_s = |S|$.

designed specifically for use with ensemble learners, into the calibration step (hereafter: Bootstrap-CP).

In contrast to fitting m neural networks on the same training data with different random weights initializations, as applied in the BANANAS framework, a different technique to build an ensemble model is via bootstrapping. Specifically, the ensemble method starts by creating multiple training datasets by resampling the available data points with replacement. In the next step, multiple models are trained on each of the bootstrapped subsets, and their predictions are aggregated to produce the single final prediction [2]. This technique offers more accurate and stable estimates than a single model and has shown superior performance in application.

Jackknife+ is a type of CP algorithm that is closely related to the leave-one-out (LOO) method [1]. Given a set of observations $\{(x_i, y_i)\}_{i=1}^n$, the idea is to fit an LOO estimator $\hat{h}_{-i}$ using all available data except for the i-th sample, and this process iterates over all individual samples. Then, the predictive interval around the i-th point is obtained by offsetting the prediction from $\hat{h}_{-i}(x_i)$ with the quantile of all LOO conformity scores. Equivalently, Jackknife+ can be viewed as a special case of CrossVal-CP when the number of folds is exactly set to $K = n$.

Jackknife+-after-boostrap [6] integrates both approaches and provides a cost-efficient wrapper by leveraging only the available bootstrapped sets and their corresponding fitted models, thereby avoiding re-fitting ensembles on each individual bootstrapped sample. [15] extends this method to online setting and proves its efficiency for time-series data. In contrast to the CP algorithms described in earlier sections, Bootstrap-CP requires

no data splitting because sampling with replacement automatically creates holdout sets. Training the bootstrap ensemble on random subsets from the full data also reduces the chance of overfitting.

Implementation of Bootstrap-CP is shown in Algorithm 4. Noteably, if a particular data point appears in all bootstrapped samples, it is then excluded from the computation of conformity scores since it has no associated LOO estimator. In Bootstrap-CP, the absolute residual is used for measuring cogvnformity, due to potentially insufficient LOO outputs for standard deviation estimation, i.e., LOO_i in Algorithm 4 might have fewer than two points. Specifically, Bootstrap-CP is only applied with the ensemble model. This concludes our experiment setups and the BANANAS-CP framework is finally evaluated under five various predictor+CP configurations.

3.3. Distribution Estimation

As described in Section 2.1.2, Bayesian optimization generally relies on a continuous posterior distribution at $X = x$ to obtain the acquisition score. Here, we denote by $F_t(x)$ the Cumulative Distribution Function (CDF) of the posterior distribution of the data point x at step t. In the context of NAS, where the target variable is assumed to be continuous and real-valued, the distribution can be represented by the inverse of its CDF, or quantile function without loss of generality, i.e. $Q_t(x) = F_t^{-1}(x)$.

In the BANANAS-CP framework, as outlined in Algorithm 1, we are able to generate calibrated quantile estimates for a finite set of discrete quantile levels. Intuitively, assuming the quantiles estimates are accurate, increasing the granularity of quantile levels should lead to a more accurate approximation of the underlying continuous distribution. However, it is computationally prohibited to estimate an infinite number of quantiles in practice, especially with significantly limited training data. Therefore, we propose an approach for constructing a continuous distribution from discrete quantile estimates with mild assumption. Specifically, the distribution estimator is defined as:

Definition Let $\{q_i\}_{i=1}^n$ be a quantile of percentile levels such that $0 < q_1 < q_2, ..., < q_n < 1$, and $\{v_i\}_{i=1}^n$ are the corresponding quantiles, i.e., $F^{-1}(q_i) = v_i$, the empirical CDF of the distributions $\hat{F}$ is constructed by applying linear interpolation between adjacent quantiles. Consequently, the Probability Density Function (PDF) of a specific interval $(v_a, v_b), a, b \in \{1, ..., n\}$ and $a < b$ is:

$$PDF(x) = \frac{q_b - q_a}{v_b - v_a}, \quad \text{if } x \in (v_a, v_b)$$

Diagnosis Analysis To assess the validity of this approach, we first perform a diagnosis analysis using synthetic datasets generated by a left-skewed Gaussian distribution, which we believe resembles the true underlying distribution of the validation performances of architectures in a search space. The experiments on the synthetic data are intended to

reflect the comparisons in a real NAS application, therefore two kinds of distribution estimators are examined: a Gaussian estimator and a linear-interpolation based quantile estimator. The analysis is repeated with different parameterizations, e.g, the number of quantiles, or the size of the samples, etc.

Table 3.1 reports the performance of three distribution estimators on synthetic datasets with various sample sizes. In particular, the linear-interpolation based quantile estimator is evaluated under 10 and 20 quantile levels, and the quantile estimates for interpolation are obtained by taking the empirical quantiles of the sample data. The estimation performance is assessed using the mean, standard deviation and Kullback–Leibler divergence [9]. For each sample size, we run 50 trials and aggregate the metrics over the trials to reduce the effects of randomness. Results in Table 3.1 indicate that the linear-interpolation based quantile estimator offers a better approximation to an asymmetric distribution than a Normal distribution, provided with sufficient data. However, a caveat is that quantile-based estimation tends to produce biased standard deviation estimates, which may lead to undesired effect.

Having considered the behaviors of the various acquisition functions employed in the real BANANAS-CP application (see Section 3.4), we additionally present several visualizations (Figure **??**) to compare the shapes of the estimated distributions with that of the true underlying distribution, thereby assuring the effectiveness of the acquisition functions.

Evaluation Metrics Within the BANANAS-CP framework, the calibration quality at a specific epoch is measured by the Root Mean Squared Calibration Error (RMSCE) [8]. Suppose $\hat{F}_t^{-1}$ is the CDF of the distribution estimated at the t-th step and y_t is the true value revealed after the estimation, consider a sequence of $\{(\hat{F}_t^{-1}, y_t)\}_{t=1}^{T}$ that represents a neural architecture search process after T epochs, the calibration error at the T-th epoch is defined as:

$$\text{RMSCE}\left(\hat{F}_1^{-1}, y_1, \ldots, \hat{F}_T^{-1}, y_T\right) = \sum_{j=1}^{m} w_j \left(p_j - \hat{p}_j\right)^2 \tag{3.8}$$

$$\text{with} \quad \hat{p}_j = \frac{\left|\left\{ y_t \;\middle|\; \hat{F}_t^{-1}(y_t) \leq p_j,\ t = 1, 2, \ldots, T \right\}\right|}{T}$$

where m represents the number of discrete quantile levels. Note that calibration errors calculated with different numbers of quantiles are not directly comparable. In general, increasing the number of quantiles tends to lead to larger calibration errors.

Table 3.1: Statistical metrics of distributions estimated by three methods on synthetic datasets with sample size ranging from 50 to 500. For each method and sample size, the mean, standard deviation, and KL divergence are reported to assess the estimation performance.

Sample Size	**Estimator**	**Mean**	**Standard Deviation**	**KL Divergence**
50	Gaussian	-0.7985	0.6005	0.5727
	Quantile ($\|q\| = 10$)	-0.7481	0.5503	0.1646
	Quantile ($\|q\| = 20$)	-0.7440	0.5536	0.2928
100	Gaussian	-0.7989	0.6145	0.6181
	Quantile ($\|q\| = 10$)	-0.7744	0.5941	0.0953
	Quantile ($\|q\| = 20$)	-0.7654	0.5798	0.1297
150	Gaussian	-0.7986	0.6100	0.5977
	Quantile ($\|q\| = 10$)	-0.7860	0.6133	0.0811
	Quantile ($\|q\| = 20$)	-0.7791	0.5950	0.0948
200	Gaussian	-0.7969	0.6162	0.6227
	Quantile ($\|q\| = 10$)	-0.7948	0.6347	0.0625
	Quantile ($\|q\| = 20$)	-0.7833	0.6050	0.0676
250	Gaussian	-0.7921	0.6127	0.6166
	Quantile ($\|q\| = 10$)	-0.7936	0.6384	0.0526
	Quantile ($\|q\| = 20$)	-0.7815	0.6084	0.0576
300	Gaussian	-0.7940	0.6140	0.6168
	Quantile ($\|q\| = 10$)	-0.8002	0.6521	0.0540
	Quantile ($\|q\| = 20$)	-0.7876	0.6153	0.0482
350	Gaussian	-0.7919	0.6131	0.6176
	Quantile ($\|q\| = 10$)	-0.8002	0.6570	0.0507
	Quantile ($\|q\| = 20$)	-0.7866	0.6188	0.0432
400	Gaussian	-0.7919	0.6131	0.6174
	Quantile ($\|q\| = 10$)	-0.8041	0.6650	0.0491
	Quantile ($\|q\| = 20$)	-0.7871	0.6215	0.0388
450	Gaussian	-0.7908	0.6114	0.6121
	Quantile ($\|q\| = 10$)	-0.8054	0.6668	0.0468
	Quantile ($\|q\| = 20$)	-0.7888	0.6240	0.0357
500	Gaussian	-0.7922	0.6081	0.5956
	Quantile ($\|q\| = 10$)	-0.8070	0.6657	0.0459
	Quantile ($\|q\| = 20$)	-0.7912	0.6219	0.0328
-	Ground Truth	-0.7939	0.6080	0.0000

3.4. Acquisition Function and Search Strategy

Finally, we describe the acquisition functions and the aquisition optimization strategies used within the BANANAS-CP framework.

3.4.1. Acquisition Functions

We consider four commonly used acquisition functions. Consistent with the notation in the previous sections, $\hat{f}(a)$ is the predicted performance of architecture a and $\hat{F}_a$ represents the CDF of the estimated distribution of $f(a)$. Then, depending on the acquisition function used, the specific acquisition score can be calculated by:

ITS: A sample is drawn from the distribution at random and its value is seen as the acquistion score for the candidate architecture a.

UCB: The acquisition score is given by $\mu + \gamma \cdot \sigma$ for a Gaussian distribution, where γ is the exploration factor. For non-Gaussian distributions, we follow [3] and generalize the function to a quantile function, i.e., $\hat{F}_a^{-1}(\gamma)$. As in the original formulation, higher values of γ promotes exploration. In our experiments, we set $\gamma = 0.75$ due to these concerns: a) the distribution of model performance is believed to be left-skewed; b) estimates of extreme quantiles are generally based on sparse observations and therefore might be less reliable. A value of 0.75 is likely to offer a reasonable trade-off between exploration and accurate estimation.

PI: The probability of improvements corresponds to $1 - \hat{F}_a^{-1}(f_{max})$, with f_{max} being the highest model performance ever observed.

EI: The expected value of improvements can be written as $\mathbb{E}[\max(0, f(a) - f_{max})]$, with f_{max} being the highest model performance ever observed.

3.4.2. Acquisition Optimization Strategy

In parallel to the settings in [14] (see Section 2.1.2), we compare three different approaches for constructing a set with 100 [1] candidates in order to compute the acquisition scores. The motivations and the specific approaches are described below:

Mutation We investigate the mutation-based search strategy because this approach demonstrates the best performance in [14]. In line with their approach, the candidates are selected by randomly changing one operation or one edge of the k best models that have been found so far, where k is a search hyperparameter with the default value of 2.

[1] We have conducted preliminary experiments using 1,000 candidate samples as well. The results indicate that increasing the sample size does not lead to significant improvements in performance. Considering the size of the search space (NAS-Bench-201), and the required computational time, the candidate set size is fixed at 100 for all subsequent primary experiments.

Random Sampling This approach is explored under the assumption that globally sampled architectures may improve the quality of the calibration process. As indicated by the name, the candidate set is created by randomly sampling architectures from the entire search space.

Dynamic This approach aims to resemble the "random + mutation" search strategy in [14]. The search process begins with random sampling until utilizing the first half of the search budget, then switches to a mutation-based strategy that progressively reduces the number of best ever-found models considered for mutation. To be precise, the number of models to be mutated decreases by 2 every 20 epoch. For instance, in a NAS task with 160 epochs, candidates are picked via random sampling for the first 80 epochs. Starting from epochs 80/100/120/140, the candidates are selected by mutating the best 8/6/4/2 models, respectively.

4. Dataset

To compare the performance of BANANAS–CP with the original BANANAS algorithms and assess the role of uncertainty calibration, we run experiments on the widely used benchmark tabular dataset NAS-Bench-201 [4].

NAS-Bench-201 is a cell-based architecture search space.

Each cell has in total 4 nodes and 6 edges. The nodes in this search space correspond to the architecture's feature maps and the edges represent the architectures operation, which are chosen from the operation set O = 1 1 conv. , 3 3 conv. , 3 3 avg. pooling , skip , zero (see Figure 1). This search space contains in total 56 = 15 625 architectures

search space defined in NAS-Bench-201 includes all possible architectures generated by 4 nodes and 5 associated operation options, which results in 15,625 neural cell candidates in total

Each of the architectures are trained on three datasets: CIFAR-10, CIFAR-100 and ImageNet-16-120.

... descriptions of each dataset.

allows research focusing on the NAS algroithms and re, which also enables standardized comparison across NAS algorithms . Besides, NAS-Bench-201 also facilitates further extension, e.g., robustness dataset.

In this work, we leverge the trained models and simply query the validation accuracy of architectures from the search space.

5. Experiments and Results

5.1. Setup

5.2. Baseline

6. Conclusion

This chapter presents the central findings of this work as well as their critical discussion. Finally, it highlights limitations and corresponding opportunities for further research.

Bibliography

[1] Rina Foygel Barber, Emmanuel J. Candes, Aaditya Ramdas, and Ryan J. Tibshirani. Predictive inference with the jackknife+, 2020.

[2] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[3] Shachi Deshpande, Charles Marx, and Volodymyr Kuleshov. Online calibrated and conformal prediction improves bayesian optimization. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 238, pages 7262–7273. PMLR, 2024.

[4] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. In *International Conference on Learning Representations (ICLR)*, 2020.

[5] Alexander Gammerman, Volodya Vovk, and Vladimir Vapnik. Learning by transduction. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 148–155, Madison, WI, USA, 1998. Morgan Kaufmann.

[6] Byol Kim, Chen Xu, and Rina Foygel Barber. Predictive inference is free with the jackknife+-after-bootstrap. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[7] Roger Koenker and Gilbert Bassett. Regression quantiles. *Econometrica: Journal of the Econometric Society*, pages 33–50, 1978.

[8] Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2796–2804. PMLR, Jul 2018.

[9] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[10] Harris Papadopoulos, Kostas Proedrou, Vladimir Vovk, and Alexander Gammerman. Inductive confidence machines for regression. In Tapio Elomaa, Heikki Mannila, and Hannu Toivonen, editors, *Machine Learning: ECML 2002*, volume 2430 of *Lecture Notes in Computer Science*, pages 345–356. Springer, 2002.

[11] Yaniv Romano, Evan Patterson, and Emmanuel J Candès. Conformalized quantile regression. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[12] David Salinas, Jacek Gołebiowski, Aaron Klein, Matthias Seeger, and Cédric Archambeau. Optimizing hyperparameters with conformal quantile regression. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

[13] Vladimir Vovk. Cross-conformal predictors. *Annals of Mathematics and Artificial Intelligence*, 74(1-2):9–28, 2015.

[14] 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*, 2021.

[15] Chen Xu and Yao Xie. Conformal prediction interval for dynamic time-series. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11559–11569. PMLR, 18–24 Jul 2021.

Acronyms

BANANAS	Bayesian Optimization with Neural Architectures for Neural Architecture Search
BANANAS-CP	BANANAS with Conformal Prediction
Bootstrap-CP	Conformal Prediction with Bootstrapping
CDF	Cumulative Distribution Function
CP	Conformal Prediction
CQR	Conformal Quantile Regression
CrossVal-CP	Conformal Prediction with Cross-validation
EI	Expected Improvements
FCP	Full Conformal Prediction
FNNs	Feedforward Neural Networks
ITS	Independent Thompson Sampling
LOO	leave-one-out
NAS	Neural Architecture Search
PDF	Probability Density Function
PI	Probability of Improvements
RMSCE	Root Mean Squared Calibration Error
SCP	Split Conformal Prediction
UCB	Upper Confident Bound

A. Program Code and Data Resources

The source code and a documentation are available at the GitHub repository: https://github.com/chengc823/Thesis. The datasets used for experiments and algorithm evaluations are sourced from the NASLib repository.

In case of access or permission issues to the private repository, please reach out at: chechen@mail.uni-mannheim.de.

B. Additional Experimental Results

Ehrenwörtliche Erklärung

Ich versichere, dass ich die beiliegende Bachelor-, Master-, Seminar-, oder Projektarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und in der untenstehenden Tabelle angegebenen Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Declaration of Used AI Tools

Tool	Purpose	Where?	Useful?
ChatGPT	Rephrasing	Throughout	+
ChatGPT	Debugging LaTeX syntax errors	Equation	+
ChatGPT	Rendering LaTeX tables from Python frame	Tables	+

Unterschrift
Mannheim, den 31.07.2025