





Predicting Phylogenetic Bootstrap Values via Machine Learning

Julius Wiegert ^{*},¹ Dimitri Höhler ¹, Julia Haag ¹, Alexandros Stamatakis ^{1,2,3}

¹Computational Molecular Evolution Group, Heidelberg Institute for Theoretical Studies, Heidelberg, Germany

²Biodiversity Computing Group, Institute of Computer Science, Foundation for Research and Technology - Hellas, Heraklion, Crete, Greece

³Institute for Theoretical Informatics, Karlsruhe Institute of Technology, Karlsruhe, Germany

*Corresponding author: E-mail: julius-wiegert@web.de.

Associate editor: Andrey Rzhetsky

Abstract

Estimating the statistical robustness of the inferred tree(s) constitutes an integral part of most phylogenetic analyses. Commonly, one computes and assigns a branch support value to each inner branch of the inferred phylogeny. The still most widely used method for calculating branch support on trees inferred under maximum likelihood (ML) is the Standard, nonparametric Felsenstein bootstrap support (SBS). Due to the high computational cost of the SBS, a plethora of methods has been developed to approximate it, for instance, via the rapid bootstrap (RB) algorithm. There have also been attempts to devise faster, alternative support measures, such as the SH-aLRT (Shimodaira–Hasegawa-like approximate likelihood ratio test) or the UltraFast bootstrap 2 (UFBoot2) method. Those faster alternatives exhibit some limitations, such as the need to assess model violations (UFBoot2) or unstable behavior in the low support interval range (SH-aLRT). Here, we present the educated bootstrap guesser (EBG), a machine learning-based tool that predicts SBS branch support values for a given input phylogeny. EBG is on average 9.4 ($\sigma = 5.5$) times faster than UFBoot2. EBG-based SBS estimates exhibit a median absolute error of 5 when predicting SBS values between 0 and 100. Furthermore, EBG also provides uncertainty measures for all per-branch SBS predictions and thereby allows for a more rigorous and careful interpretation. EBG can, for instance, predict SBS support values on a phylogeny comprising 1,654 SARS-CoV2 genome sequences within 3 h on a mid-class laptop. EBG is available under GNU GPL3.

Key words: bootstrap support, machine learning, phylogenetics, uncertainty estimation.

Introduction

Inferring phylogenetic trees under the maximum-likelihood (ML) criterion is time—and resource-intensive, as the number of possible tree topologies increases super-exponentially with the number of taxa under study. As a consequence, tree search algorithms, as implemented in RAXML-NG (Kozlov et al. 2019), conduct their search for a best-known, yet not necessarily globally optimal, tree topology via a plethora of distinct heuristics. As there is no guarantee that the tree search will converge to the globally optimal tree, subsequent analyses to quantify its uncertainty are necessary. Such uncertainty analyses constitute an integral as well as routine component of current phylogenetic analysis pipelines (Kapli et al. 2020). The most common approach to quantify uncertainty is to infer various types of inner branch support values. The still most common technique to calculate branch support values on a resulting phylogeny under ML is the Standard, non-parametric classic Felsenstein bootstrap support (SBS)

(Felsenstein 1985). The SBS randomly samples the alignment sites of the multiple sequence alignment (MSA) with replacement to create a set of replicate MSAs (called bootstrap replicates). On each such replicate, one subsequently infers a respective ML tree. Thus, the SBS yields a set of bootstrap replicate ML trees. This SBS procedure is time- and resource-consuming, due to the high computational cost of conducting a phylogenetic tree inference on each replicate. Note that typically 100 to 500 replicate trees need to be inferred to obtain stable support values (Pattengale et al. 2010).

To alleviate this computational bottleneck, a plethora of alternative, faster methods to infer branch support have been proposed. For example, Stamatakis et al. (2008) propose the rapid bootstrap (RB) as part of the phylogenetic inference tool RAXML (Stamatakis 2014) as a faster alternative to the SBS. RB uses a heuristic approach that implements a more superficial ML tree search to approximate the SBS values and reduce computational costs.

Received: April 23, 2024. **Revised:** August 28, 2024. **Accepted:** September 15, 2024

© The Author(s) 2024. Published by Oxford University Press on behalf of Society for Molecular Biology and Evolution.

This is an Open Access article distributed under the terms of the Creative Commons Attribution-NonCommercial License (<https://creativecommons.org/licenses/by-nc/4.0/>), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited. For commercial re-use, please contact reprints@oup.com for reprints and translation rights for reprints. All other permissions can be obtained through our RightsLink service via the Permissions link on the article page on our site—for further information please contact journals.permissions@oup.com.

Open Access

On multiple large datasets (Stamatakis et al. 2008) show that RB support values are highly correlated with the SBS values (Pearson correlation between 0.92 and 0.99). The Ultrafast Bootstrap (UFBoot) (Minh et al. 2013) and its current version UFBoot2 (Hoang et al. 2018) interweave parametric (substitution model-dependent) and non-parametric aspects. While the tree space sampling of UFBoot2 is parametric, it deploys a nonparametric bootstrap sampling of the MSA. UFBoot2 yields easy-to-interpret, unbiased branch support values, the UltraFast bootstrap support (UFBS). According to the authors' experiments, UFBoot2 is extremely fast, as, on median, it is 778 times faster than SBS and 8.4 times faster than RB.

Both RB (optionally) and UFBoot2 employ an iterative approximation of their branch supports until they either meet a stopping criterion or reach a maximum number of iterations. Thus, the runtimes of these methods can vary, since they depend on the input MSA, which determines if and when the support value calculations will converge.

Anisimova and Gascuel (2006) propose an alternative definition of branch support based on a parametric method for branch support estimation: the approximate likelihood ratio test (aLRT). The aLRT compares the two best nearest-neighbor interchange (NNI) moves at each inner branch via an LRT test, to calculate a branch support value. To accelerate the NNI likelihood evaluation, aLRT only optimizes the branches adjacent to the branch of interest. As aLRT is parametric, it can be sensitive to substitution model violations. Substitution model violations occur, for example, when we choose a substitution model that is too simple for the data at hand. To correct for those violations, Guindon et al. (2010) propose the Shimodaira–Hasegawa-like (SH-like) aLRT, which constitutes a non-parametric version of the aLRT. The aLRT and SH-like aLRT only focus on local perturbations of the given ML topology on which they calculate supports. This can induce overconfidence in branches if there exist other highly likely, yet topologically substantially distinct tree topologies (Guindon et al. 2010). One recent example of such a tree space with a multitude of topologically highly distinct, yet almost equally likely tree topologies is a phylogeny of SARS-CoV2 genome sequences (Morel et al. 2020).

Despite the availability of these tools, SBS remains an important approach for assessing branch support (Brandis 2021; Ahmed et al. 2022; Steck et al. 2022). Guindon et al. (2010) propose to combine the SBS with the SH-like aLRT to obtain a holistic estimate of branch robustness. UFBoot2 is less vulnerable to severe model violations than UFBoot (Hoang et al. 2018). Nonetheless, UFBoot2 still requires an additional step to assess such violations. SBS is inherently robust against model violations, as it is nonparametric.

Here, we present the educated bootstrap guesser (EBG), a machine learning-based approach for predicting SBS values. Predicting SBS values on 234 empirical MSAs with EBG is on average 9.4 ($\sigma = 5.5$) times faster with respect to time-to-completion compared to UFBoot2. Based on

these experiments, we show that EBG can also predict whether a specific branch will exceed a particular SBS threshold (e.g. $t = 80$) in a 1,000 replicate SBS run with a balanced accuracy (BAC) of at least 0.91. EBG is available as an open-source command line tool on GitHub github.com/wiegertj/EBG.

Materials and Methods

Phylogenetic Bootstrap

On each replicate, the SBS infers a corresponding ML tree, which yields a set of replicate trees. We can use the set of replicate trees to either construct a consensus tree or to map confidence values onto a reference tree (typically the best-known ML tree) as SBS values. The SBS value for a specific inner branch (also referred to as nontrivial split or bipartition) is the frequency of occurrence of this branch in the replicate tree set. Common representations of the SBS values are percentage values between 0 and 100, or fractions between 0 and 1. In the following, we represent SBS values as percentage values between 0 and 100.

Problem Formulation

We address the challenge of predicting the SBS values via two distinct steps: regression and classification. The EBG regressor directly predicts the respective SBS values for *all* inner branches of a given ML tree. In the classification approach, EBG then predicts the probability of the SBS value of each single inner branch exceeding a given SBS threshold using the regressor output as input. This classification step is based on the SBS interpretation by Felsenstein and Kishino (1993). The authors propose to interpret the quantity of one minus the SBS value as a *P*-value for the null hypothesis of the branch *not* forming part of the true tree. However, Susko (2009) shows that one minus the SBS is too conservative as a *P*-value. Given an SBS value > 95 , the probability of the branch not being in the true tree is substantially lower than 5%. Their experiments suggest that, depending on the characteristics of the true tree, an SBS between 70 and 85 yields a 5% false positive (FP) bound. Consequently, we use this SBS range for our classification step. More specifically, we predict the probability for each single inner branch *i* to exceed a specific SBS threshold *t*, that is, $\text{SBS}_i > t$ with $t \in \{70, 75, 80, 85\}$.

With our novel EBG tool, we address both the regression and the classification challenge. In addition, we estimate the uncertainty of the resulting predictions.

Training Data

Trost et al. (2024) demonstrate that machine learning algorithms can easily distinguish between simulated and empirical MSAs with high accuracy and conclude that sequence simulations do not fully capture all characteristics of empirical MSAs. Consequently, we exclusively used empirical MSAs to train EBG. We obtained the empirical MSAs from TreeBASE (Piel et al. 2009). As TreeBASE only contains MSAs of published studies, we assume that

they are representative of commonly analyzed datasets by practitioners.

We used 1,496 MSAs (93% DNA and 7% amino acid [AA]) for training and evaluating EBG. In addition, we randomly selected 234 additional MSAs for our final comparison with RB, UFBoot2, and SH-like aLRT. We noticed that approximately half of the MSAs in TreeBASE contain at least two exactly identical sequences, and therefore decided to remove all duplicate sequences before training EBG. We selected the MSAs based on the Pythia difficulty (Haag et al. 2022). For a given MSA, Pythia quantifies the difficulty of the phylogenetic analysis under the ML criterion. To obtain a comprehensive representation of the Pythia difficulty spectrum in ML tree inference, we therefore predicted the Pythia difficulty for each MSA in TreeBASE using Pythia. Subsequently, we selected MSAs representing distinct Pythia difficulty levels to cover both, easy, and challenging MSAs.

For each MSA, we inferred 100 ML trees under the GTR +G substitution model using RAXML-NG (Kozlov et al. 2019) and used the one with the highest log-likelihood as the ML tree. To obtain SBS “ground truth” values as a training target for EBG, we performed one SBS run with 1,000 replicates for each MSA using RAXML-NG. Our final training dataset comprises approximately 80,000 inner branches and their corresponding SBS value. Ändeurng

Feature Engineering

To predict the SBS values, EBG computes a plethora of MSA features but also extracts features from the respective best-known ML tree, including model parameter estimates. EBG uses a total of 23 features for the prediction (Table 1).

The majority of features are extracted from a set of parsimony starting trees (henceforth simply referred to as parsimony trees) that EBG infers using RAXML-NG. RAXML-NG infers parsimony starting trees via a randomized stepwise addition order algorithm (—start-option). The development of Pythia already showed that by using the computationally substantially less expensive parsimony trees, we can accurately predict features of the ML tree space (Haag et al. 2022). Due to the high prediction accuracy of Pythia, we therefore expected that parsimony-based features will also be useful for predicting SBS values.

We calculated a set of 12 features based on parsimony trees. Those 12 features are subdivided into parsimony support (PS) and parsimony bootstrap support (PBS) features. Figure 1 provides an overview of the feature computation and its inputs.

The PS is the frequency of occurrence of an inner branch in 1,000 parsimony starting trees that EBG infers on the original MSA. According to exploratory experiments (see supplementary section 7, Supplementary Material online), we set the number of inferred parsimony trees to 1,000, as more than 1,000 did not substantially improve prediction performance.

The PBS employs a procedure that is highly similar to the SBS and uses replicate MSAs. The only difference is,

Table 1. Overview of the features subset used in EBG

Feature
Parsimony bootstrap support (PBS)
Std. PBS parents
Mean PBS parents
Std. PBS children
Min. PBS children*
Max. PBS children*
Skewness PBS
Mean Robinson–Foulds distance parsimony bootstrap trees
Parsimony support (PS)
Min. PS children
Max. PS children
Max. PS children*
Min. PS children*
Normalized ML branch length
No. of child inner branches
Mean parsimony mutation per side (PMS)
Coefficient of variation PMS
Max. PMS
Skewness PMS
ML tree branch length
Branch number (ordered by level-order traverse)
Branch length ratio bipartition
Mean closeness centrality bipartition ratio

*: weighted by branch length.

that PBS computes a parsimony starting tree for each bootstrap replicate, instead of inferring an ML tree, yielding the computation substantially faster. To capture the variance of the respective bootstrap tree space, EBG also uses the mean normalized Robinson–Foulds (nRF) (Robinson and Foulds 1981) distance between all PB trees as a feature. Again, based on exploratory experiments, inferring more than 200 PBs does not improve predictor performance (see supplementary section 7, Supplementary Material online).

In theory, we could decrease the number of trees as a function of the Pythia difficulty score, since the tree space is less complex for lower Pythia difficulties. Hence, for lower Pythia difficulties, a smaller number of parsimony trees might be sufficient to approximate the SBS values. However, because this might introduce additional uncertainty in the prediction process, EBG keeps the number of parsimony trees for both the PBS and the PS features fixed.

We expected that the P(B)S values of branches adjacent to a focal inner branch of interest are indicative of its SBS value. Therefore, we also included summary statistics for the P(B)S values of respective child and parent branches as features. Additionally, we computed summary statistics over the per-site parsimony mutation count scores (PMS) as an additional group of features. Finally, EBG uses the mean closeness centrality (Bavelas 2005) of the two subtrees connected to the focal inner branch. The closeness centrality quantifies how densely connected the tree’s nodes are to each other. By taking the closeness centrality ratio of those subtrees, we aim to capture if the branch connects two subtrees of different densities. Another feature, that we refer to as the branch length ratio bipartition,

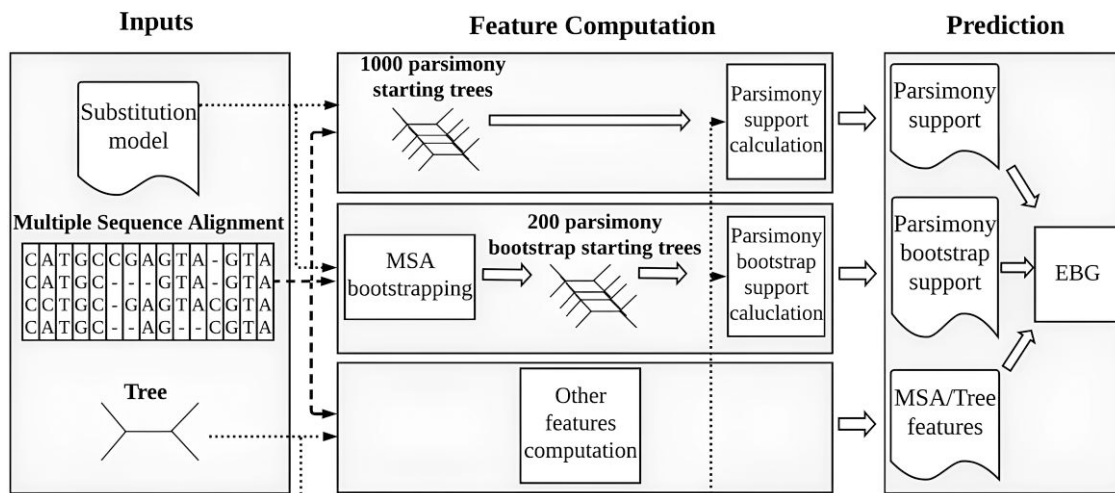


FIG. 1. Overview of the EBG feature generation and prediction.

represents the ratio between the sums of branch lengths in the two subtrees induced by the focal branch.

The set of 23 features (see Table 1) that EBG uses is a subset of a larger set of over 150 features we experimented with. The supplementary section 2, Supplementary Material online comprises a detailed description of all features. We reduced this initial set of 150 features to 23 features via recursive feature elimination (Guyon et al. 2002) and a scikit-learn random forest (Pedregosa et al. 2011). The aim was to determine a good trade-off between the number of features and EBG performance for the following reasons. First, we strive to avoid unnecessary computations. Second, fewer features yield the predictor more interpretable. Third, we aim to prevent overfitting our predictor to the training data. This can occur when training a predictor with too little training data and too many redundant or nonpredictive features (Hawkins 2004). In the case of overfitting, the predictor will not generalize well to unseen inputs.

Predictor and Performance Metrics

According to preliminary experiments (cf. supplementary section 6, Supplementary Material online), the best machine learning model choice for SBS prediction (both regression and classification) is the light gradient-boosting model (LightGBM) (Ke et al. 2017) tree-based boosting ensemble framework. To quantify the confidence of the EBG regression, we estimate the prediction uncertainty using quantile regression (Koenker and Bassett 1978). We deploy this approach to estimate the conditional quantiles of the SBS value. By training the model to predict the 5th and 10th quantile of the SBS value, EBG provides lower bounds for the SBS value at 5% and 10%.

We optimized the hyperparameters of all models in 100 trials using the Optuna (Akiba et al. 2019) hyperparameter optimization framework. Throughout the training process, we ensured that the TreeBASE MSAs we used were either

exclusively contained in the training set or the test set. This guaranteed, that the inner branches of the respective ML tree are never split between the training and test set. Thereby, we obtained an estimate of the predictors' performance on entirely new, that is, unseen trees. We evaluated the performance of EBG using two distinct sets of metrics, one for the regression, and one for the classification step. In the following, we briefly describe both sets of metrics (see supplementary sections 3 and 4, Supplementary Material online for further details on their computation).

Botchkarev (2019) proposes a regression metric typology, which we used to evaluate the EBG regressor. The typology describes regression performance metrics based on aggregation methods, distance measures, and normalization methods. Normalization is only necessary if we compare multiple predictions of different scales, which does not apply to EBG. We chose both, mean, and median as aggregation methods since the mean is sensitive to outliers, while the median is not. Additionally, we selected three distance measures: the squared distance, the difference, and the absolute difference between the SBS and EBG values. The squared distance between the SBS values and the EBG prediction is sensitive, whereas the difference and the absolute difference are robust to outliers. This results in the following four metrics for evaluating the EBG regressor: the mean bias error (MBE), the mean absolute error (MAE), the median absolute error (MdAE), and the root mean squared error (RMSE). We used the MBE to evaluate whether the predictor is biased toward over- or underestimation. The MAE provides an outlier-sensitive, average performance of the predictor. The MdAE yields an outlier-insensitive performance metric. Finally, we used the RMSE as a measure that penalizes larger prediction errors more harshly.

The EBG classifier predicts class probabilities for the binary classification task, that is, the probability that an SBS value is below or above a specific SBS threshold. We set the decision boundary for the binary classification to 0.5.

We evaluated the prediction accuracy of the EBG classifier using four metrics: the accuracy (Acc), the BAC, the F1-score (F1), and the area under the curve (AUC) of the receiver operating characteristic (ROC) curve. The Acc is easy to interpret and sensitive to class imbalances. The BAC is equivalent to the Acc with a weighting that is inversely proportional to the class prevalence. Therefore, the BAC is insensitive to class imbalances. The F1 is the harmonic mean between precision and recall, thus it penalizes FP (erroneously predicting that the SBS value surpasses the SBS threshold) and false negatives (FN) as being equally unfavorable. The F1 is robust against class imbalances. In contrast to Acc and F1, the AUC allows for a classification performance evaluation without the need to set a class probability decision boundary, since it relies on the raw class probabilities.

Results

We evaluated the performance of the EBG regressor and classifier using different splits of our curated dataset of 1,496 TreeBASE MSAs into training and testing MSAs. For both the EBG regressor and classifier, we further tested the usage of EBG's uncertainty measures for quantifying the reliability of its predictions. For our analysis, we refer to the usage of the uncertainty measures for EBG's predictors as *filtering* and the respective uncertainty thresholds as *filter values*. We also compared EBG with RB, UFBoot2, and SH-like aLRT regarding prediction quality, and we compared EBG against its fastest competitors UFBoot2 and SH-like aLRT in terms of time-to-completion and accumulated CPU time. Furthermore, we analyzed the importance of the prediction features for EBG.

EBG Regressor Performance Evaluation

The EBG regressor predicts three values. In addition to the central SBS point estimate, EBG provides two SBS predictions that correspond to two lower bounds. One lower bound has a 5%, the other a 10% predicted probability that the SBS value is below the respective bound. On a random subset of 232 MSAs, EBG's central SBS estimate is highly correlated, showing a mean Pearson correlation of $\mu = 0.91$, with the actual SBS values. Note that we calculated the mean Pearson correlation using Fisher's z-transformed Pearson correlations (Dunlap et al. 1998). For a more detailed view, we refer to [supplementary section 9, Supplementary Material](#) online. For the evaluation of the EBG regressor's central SBS point estimate, we randomly sampled 20% of the 1,496 MSAs as a holdout testing dataset and trained EBG on the remaining 80%. [Table 2](#) summarizes the results of 10 such random holdouts, showing the mean and standard deviation of EBG's performance across all 10 holdouts.

To assess the predictive power of EBG, we defined a baseline SBS value prediction based on the PBS of 200 replicates. If a branch is in the ML tree, but not in the PBS replicate tree set, the baseline prediction is zero, as this resembles the behavior of the SBS procedure. EBG

Table 2. EBG regression performance for 10 repeated random holdouts against a parsimony bootstrap support of 200 replicate trees as the baseline

Metric	EBG ($\mu \pm \sigma$)	Baseline ($\mu \pm \sigma$)
MBE	0.6 \pm 0.3	0.1 \pm 0.5
MAE	8.3 \pm 0.2	13.8 \pm 0.4
MdAE	5.0 \pm 0.1	8.6 \pm 0.5
RMSE	12.8 \pm 0.2	20.5 \pm 0.5

To infer those replicate trees, we first obtained 200 replicate MSAs by sampling the original MSA in a site-wise manner with replacement. Subsequently, we inferred the corresponding parsimony (starting) tree using RAxML-NG to calculate the 200 replicate trees.

outperformed this baseline across all four metrics. As the MBE indicates, the regressor exhibits no substantial systematic bias with respect to either over—or underestimating SBS values. The difference between MAE and MdAE, along with the higher RMSE value, suggests deviations of varying size between the EBG prediction and the true SBS value. The lower bound predictions of EBG can serve as a means to establish a bound for this prediction error and thus enable filtering for certain predictions.

[Figure 2](#) illustrates, how the proximity of the lower bound predictions to the median prediction can be used to constrain the MdAE for the EBG median regression predictions. This approach effectively mitigates prediction uncertainty by filtering. As the lower bound predictions approach the median prediction, the MdAE decreases. Thus, the inspection of the difference between the lower bound and the median SBS predictions limits and quantifies the prediction uncertainty.

We investigated the performance of EBG on a four-taxon tree from the Felsenstein zone (Huelsenbeck and Hillis 1993), a tree on which maximum parsimony typically fails to reconstruct the correct tree whereas ML recovers it. We used that tree to simulate an MSA with 1,000 sites under the Jukes–Cantor (JC) model using AliSim. EBG underestimates the support value (72) of the true branch. However, we are able to detect its uncertainty via the 5% lower bound filter value ≤ 23 . Besides that, recent results on ML reconstruction accuracy in the Felsenstein zone show that the accuracy as determined via large-scale experiments is as low as 70% (Leuchtenberger et al. 2020). This further indicates that even for ML methods, the Felsenstein zone remains difficult terrain.

EBG Classifier Performance Evaluation

In analogy to the EBG regressor, we evaluated the EBG classifier based on 10 repeated holdout sets of 20%. [Table 3](#) summarizes the resulting performance metrics for varying SBS thresholds t .

As baseline performance, we used the common SBS thresholds of $t: = 80$ (Koenen et al. 2020; Brandis 2021; Cruaud et al. 2021) on the PBS of 200 replicates. The baseline again is zero for all branches, that are present in the ML tree, but not in the PBS replicate trees. EBG outperformed the baseline for every metric.

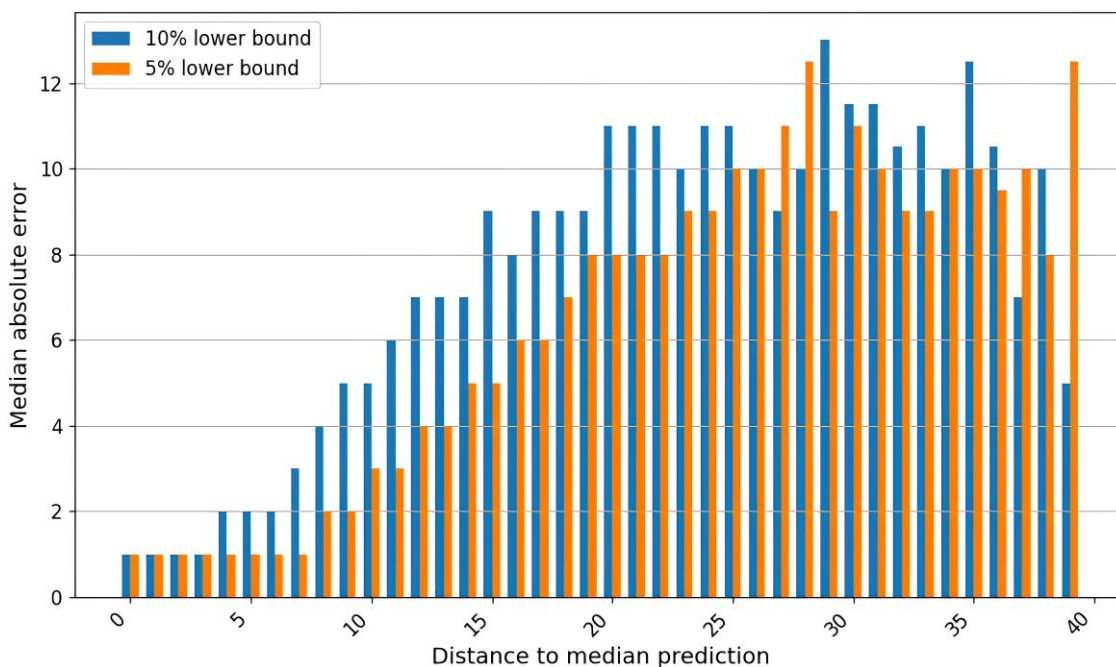


Fig. 2. Relationship between the difference of the lower bound predictions to the median prediction and their influence on the MdAE.

Table 3. EBG classifier performance for different decision boundaries t

Metric	$t = 70$	$t = 75$	$t = 80$	$t = 85$	baseline
BAC	0.92 ± 0.00	0.91 ± 0.00	0.91 ± 0.00	0.92 ± 0.00	0.85 ± 0.01
AUC	0.97 ± 0.00	0.98 ± 0.00	0.98 ± 0.00	0.98 ± 0.00	0.85 ± 0.01
F1	0.90 ± 0.00	0.90 ± 0.00	0.89 ± 0.00	0.89 ± 0.00	0.82 ± 0.01

We indicate the mean and standard deviation of 10 repeated random holdouts against a baseline consisting of the PBS of 200 replicates for $t = 80$. We decided to use 200 replicates as this is the number of replicates that EBG uses for its PBS feature computation.

In analogy to the lower bound prediction for the EBG regressor, EBG also provides a prediction uncertainty measure for its classifier for filtering out uncertain predictions. As the EBG classifier solves a binary classification problem, with one class defined as $SBS > t$ and the other as $SBS \leq t$, we can leverage the Shannon entropy (Shannon 1948) of the two class probabilities to obtain a prediction uncertainty measure $u \in [0, 1]$. Here, $u = 0$ represents absolute certainty, and $u = 1$ corresponds to absolute uncertainty. For the EBG classifier, we use u for filtering out uncertain predictions. Figure 3 provides an overview of the relationship between EBG classifiers with different SBS thresholds t and their Acc with increasing prediction uncertainty filter values. In particular for smaller uncertainties, a high-class imbalance implies an improved classification within the uncertainty interval of concern. Therefore, we decided to not compensate for class imbalances by using the BAC and used the Acc for this analysis instead. For cases with low $u \in [0.1, 0.3]$, the prediction Acc consistently remains at or above 90% across all SBS thresholds t . For moderate $u \in [0.4, 0.6]$, the Acc typically falls within the 80% to 90% range. We only observe prediction accuracies below 70% for $u > 0.8$.

In practical terms, when using EBG, we have the flexibility to determine an acceptable filter value for uncertainty to attain confident predictions. Thereby, users can tailor the approach to their specific needs and preferences.

Performance in the Critical Support Range

Thus far, we presented EBG's overall performance. However, not all support value ranges are equally critical for practical interpretation. Therefore, we analyzed EBG's performance in the critical SBS range between 60 and 95. We observe that EBG's regressor prediction error increases up to an MAE of 10.6 and an MdAE of 7.1, respectively. A detailed error distribution analysis is provided in [supplementary section 10, Supplementary Material](#) online.

To assess the performance of the EBG classifier in the critical range, we use the fraction of FP according to SBS thresholds t . The FP rate represents the fraction of predictions where EBG incorrectly classifies the branch support to exceed t . Therefore, the FP rate is of great relevance to practitioners. Table 4 summarizes the results.

Table 4 once more emphasizes the importance of EBG's uncertainty prediction as a filtering tool. When discarding the uncertain predictions, we can halve EBG's FP rate. Overall, EBG's regressor exhibits a larger error in the critical

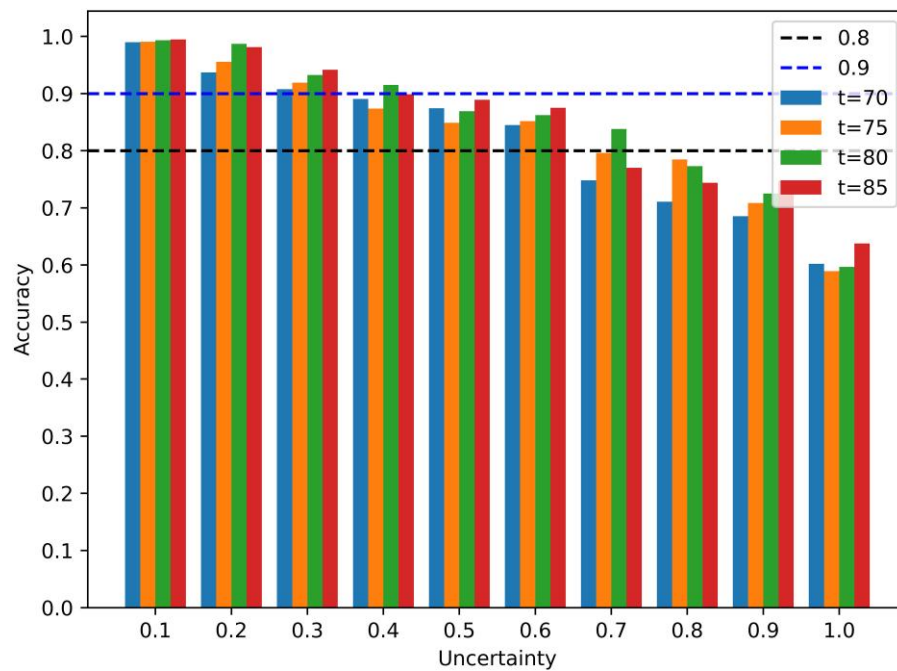


Fig. 3. Relationship between the prediction Acc of EBG classifiers and their prediction uncertainty filter values for varying SBS thresholds t .

Table 4. FP rate of the EBG classifier in the critical SBS range (subscript c) between 60 and 95 on a holdout set of 20%

Threshold	FP rate	FP rate*	FP rate _c	FP rate _c *
70	0.11	0.04	0.33	0.19
75	0.10	0.03	0.27	0.13
80	0.08	0.02	0.20	0.08
85	0.07	0.02	0.16	0.07

Column "FP rate*" denotes an uncertainty filtering of the predictions with a filter value of $u \leq 0.6$.

SBS interval between 60 and 95 compared to the entire SBS value range. Therefore, we highly recommend using the EBG classifier, as its uncertainty filtering substantially reduces the crucial FP rate.

In summary, EBG's regressor exhibits a larger error in the critical interval between an SBS of 60 and 95 compared to the whole SBS range. Therefore, we recommend using the EBG classifier instead, as its uncertainty filtering effectively bounds the crucial FP rate.

Feature Importances

Table 5 lists the five most important features of the EBG regressor. The feature importance quantifies to which extent a feature contributes to achieving an improved prediction during training.

According to the feature importances of EBG, the PBS feature with a feature importance of 82.2% is by far the most important one for predicting SBS values. We provide further insight into the relationship between EBG's prediction and the PBS in [supplementary section 11, Supplementary Material](#) online. We interpret the substantial importance of the PBS features with an analogy to ensemble methods in machine learning. These methods aggregate numerous weak learners, to create a robust,

Table 5. Overview of the five most important features that EBG uses for the prediction and their respective importance in percent

Feature	Importance in %
PBS	82.2
PS	3.1
Normalized branch length	2.0
No. of child inner branches	1.7
Skewness PBS	1.5

strong one. Similarly, EBG uses the contributions of multiple "weak" parsimony inferences, to obtain a precise estimate of the SBS values. However, using only the PBS or exclusively the PS values results in worse performance compared to EBG as indicated by the baselines used for evaluation (Tables 2 and 3 and [supplementary section 7, Supplementary Material](#) online). This aligns with the results of [Buckley and Cunningham \(2002\)](#). On a comparatively small set of phylogenies, the authors showed that PBS alone constitutes the worst support predictor of true branches compared to SBS estimates under models of varying complexity. While PBS is the most important EBG feature, it is the combination with the remaining features that enables an accurate SBS prediction. See [supplementary section 2, Supplementary Material](#) online for a more detailed overview of all prediction features and their importance.

Performance Comparison with UFBoot2, SH-Like ALRT, and RB

EBG, UFBoot2, and SH-like aLRT branch support values all have different interpretations. EBG approximates the SBS which is conservative, whereas UFBoot2's UFBS values are unbiased ([Minh et al. 2013](#)). SH-like aLRT is also

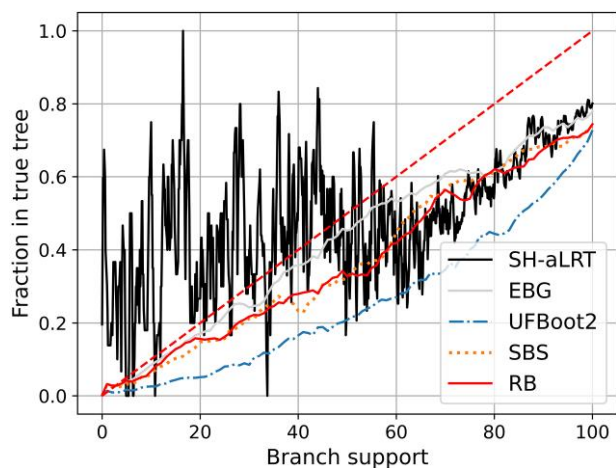


Fig. 4. Moving average with window size five of the fraction of branches in the true tree for all branch support values.

conservative but not in the same way as SBS, reasonable thresholds for SH-like aLRT can be between 0.8 and 0.9 (Guindon et al. 2010). Therefore, to compare all three branch supports against each other, we required a ground truth phylogeny. Consequently, we performed the following performance comparison using simulated MSAs. We simulated a total of 979 DNA MSAs without gaps (i.e. without simulating indel events) along empirical ML trees inferred on TreeBASE MSAs using AliSim (Ly-Trong et al. 2022) under the GTR+G model. The corresponding tree (true tree) of the simulated MSAs served as the ground truth for our experiment.

On all simulated datasets, we then performed one ML search with default parameters per simulated MSA using RAXML-NG and used the resulting ML tree as input tree for EBG. We added SBS and RB to the evaluation for further comparison and mapped SBS and RB values to the same ML tree.

We also computed SH-like aLRT support values using IQ-TREE2 (Minh et al. 2013); we provide all commands we used in [supplementary section 1, Supplementary Material](#) online. We then mapped SH-aLRT and UFBS support values to the ML tree computed by the corresponding IQ-TREE2 command.

Therefore, our experimental setup also differs from the evaluation of UFBoot(2) (Minh et al. 2013; Hoang et al. 2018) as the authors used the set of unique branches of all bootstrap trees for further evaluation. Instead, we use the branches and corresponding support values of the single ML tree, either inferred using RAXML-NG or IQ-TREE2. We decided to alter the procedure in this way since EBG is not computing a set of bootstrap replicate trees. In addition, this specific experimental setup represents one common use case of the respective support methods on empirical data. The other common use case, that is, constructing a consensus tree on the set of SBS replicates can not be assessed since EBG, or the SH-aLRT test, for instance, does not generate a tree set. Therefore, we ended up with one ML tree per tool with the corresponding

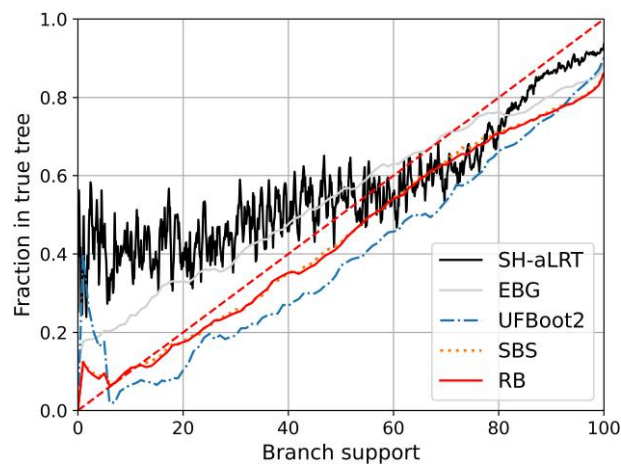


Fig. 5. Moving average with window size five of the fraction of branches in the true tree for all branch support values using the PANDIT-based DNA simulations used for the evaluation of the original UFBoot method.

support values. Note that, the ML trees inferred on the simulated datasets by RAXML-NG and IQ-TREE2 might be topologically distinct trees. In fact, we observe substantial nRF distances between the ML trees inferred by RAXML-NG and IQ-TREE2 ($\mu = 0.30$, $\sigma = 0.28$), yet approximately 20% of the inferred RAXML-NG and the corresponding IQ-TREE2 trees are topologically identical.

We used those trees to compute the fraction of branches in the true tree for each branch support value. We emphasize that SH-aLRT was not designed to be interpreted in this manner (Guindon et al. 2010). However, for the sake of completeness and comparability between the different fast support value inference methods, we decided to include them in our analyses. Figure 4 summarizes the results of our analyses. An ideal branch support measure would yield the unbiased probability of the branch being in the true tree (dashed, red line). In our experiments, all three tools, the estimation of the true branch probability is too liberal. As already observed by Minh et al. (2013), low SH-like aLRT values (<50) are not informative with respect to the true probabilities. For larger values, the SH-like aLRT behaves analogously to EBG. EBG is fairly unbiased for low support values (<60). For branch supports >60 , it tends to overestimate the true probability of the branches present in the true tree. In this experiment, UFBoot2 overestimates the true branch probability the most. Besides all three tools being overconfident in predicting the true support, EBG is the closest to the ideal branch support value line.

We also compared the performance of the different tools as a function of the Pythia difficulty of the simulated MSAs. We computed the BAC for specific branch support thresholds and compared them across all tools. The results suggest that EBG is best able to handle varying MSA difficulties. We provide a detailed analysis in [supplementary section 5, Supplementary Material](#) online.

Our results using the MSAs simulated by AliSim differ substantially from the evaluation results in the original

UFBoot experiments that rely on simulations from the PANDIT (Whelan et al. 2003) database. Therefore, we also replicated the experimental setup deployed to evaluate tool performance on the set of PANDIT-based DNA simulations used in the evaluation of UFBoot (Fig. 5).

While the results on those MSAs are closer to the ideal branch support measure, they still differ from the results presented by the authors of UFBoot. We assume that one reason for the discrepancy is the aforementioned differing experimental setup (focus on mapping support values on tree versus focus on all bipartitions). Furthermore, the differences between our AliSim-based simulation results (Fig. 4) and those of the PANDIT-based simulations (Fig. 5) are most likely due to the pronounced dissimilarity of the respective underlying Pythia difficulty distributions over the datasets (Fig. 6). On average, the PANDIT-based DNA simulations exhibit a very low Pythia difficulty ($\mu = 0.09$, $\sigma = 0.14$), whereas the AliSim simulations better cover the phylogenetic inference difficulty range ($\mu = 0.33$, $\sigma = 0.24$). Furthermore, we observe that for the PANDIT-based data 74% of the mapping target trees inferred by RAXML-NG and the corresponding IQ-TREE2 tree are topologically identical, a consequence of the low Pythia difficulty scores. For more details, we refer to [supplementary section 10, Supplementary Material](#) online.

As an additional performance assessment, we analyzed the TP and FP rates for different support thresholds on our simulated datasets. Ideally, support inference methods should attain a high TP rate and minimize the FP rate. We used the EBG classifier for this comparison as it performs better in the critical support range. Table 6 summarizes the results. To yield different support inference tools comparable, we apply the same support threshold to all of them. We nonetheless emphasize again that SH-aLRT was not designed to be interpreted in this way, but that we included it for the sake of completeness.

Particularly for stricter filtering (smaller uncertainty filter values), the EBG classifier yields the best performance in terms of reduced FP rate compared to its competitors at the cost of not providing predictions for all branches. The FP and TP rates both drop to zero in this experiment for uncertainty filter values $u \leq 0.16$. We interpret this as EBG being particularly confident about cases where the respective branch is not in the true tree, resulting in an exceptionally low FP rate. However, this conservative behavior results in a TP rate of zero for 7% of the true branches in this case, when considering uncertainty filter values of $u \leq 0.16$ or smaller. The EBG classifier even outperforms SBS with respect to both, TP, and FP rates when using an uncertainty filtering with an uncertainty filter value $u \leq 17$. UFBoot2's high TP fraction comes at the cost of the highest FP fraction. Interestingly, RB and SBS perform equally well in this experiment. We provide an analogous analysis for the EBG regressor in [supplementary section 12, Supplementary Material](#) online.

Finally, we directly compared EBG with RB on empirical MSAs. This is possible since RB is highly correlated with the SBS values (Stamatakis et al. 2008). We randomly selected

234 MSAs (20% AA, 80% DNA) from TreeBASE and computed the ground truth SBS for each branch based on 1,000 bootstrap replicates using RAXML-NG. Table 7 summarizes the respective classification and regression metrics for EBG and RB.

Based on our experimental findings, RB achieves the highest Acc when predicting SBS values. EBG's prediction performance falls short compared to RB, as evidenced by a higher MdAE (RB: 2.0, EBG: 6.0) and a lower BAC (RB: 0.96, EBG: 0.89). However, if we also use EBG's uncertainty filtering, the performance becomes comparable, in particular for the EBG classifier.

Note that EBG* in Table 7 summarizes the results for an uncertainty filtering of the predictions we conducted for both, the regression, and classification tasks. In the regression scenario, our focus is on branches where we expect the MdAE to be less than or equal to 8 (as Fig. 2 depicts), that is, a 5% filter value of ≤ 23 . This uncertainty filtering results in the exclusion of 28% of predictions that we deem as being too uncertain for consideration. For classification, we restrict our attention to predictions with a filter value of $u \leq 0.7$ (as Fig. 3 depicts). This leads to excluding 21% of the predictions, that we consider as being too uncertain. While this approach may not provide predictions for every branch, it effectively constrains prediction errors. It represents a trade-off between the number of predictions that we consider to be trustworthy and the level of certainty of those predictions.

In addition to the above accuracy analyses, we conducted a comparison of the time-to-completion of EBG with UFBoot2 and the SH-like aLRT (using IQ-TREE2) as fastest competitors on empirical datasets. The SBS computation with RAXML-NG, UFBoot2 as well as the aLRT implementation of IQ-TREE2 can use multiple threads. Since the independent parsimony tree inferences necessary for the feature computation of EBG can also be parallelized straightforwardly, we performed our benchmark on a reference machine using multiple threads. This reference machine is equipped with an Intel Xeon Platinum 8260 Processor (48 physical cores, 2.4 GHz) and 754 GB memory. RAXML-NG and IQ-TREE2 provide the option to automatically determine the optimal number of threads for a given MSA, and we used this feature in both tools with up to a total of 60 threads. Figure 7 summarizes the results of the benchmark on the 234 empirical TreeBASE MSAs we used for the EBG/RB comparison. For the sake of simplicity, we define MSA size as the product of the number of sequences times the number of site patterns (unique MSA sites). Furthermore, we separately depict run times for AA (dashed lines) and DNA datasets (straight lines) to assess potential differences between data types. Since EBG requires an existing phylogenetic tree and substitution model parameters as input, we included the inference time of adaptive RAXML-NG (Togkousidis et al. 2023) in the time-to-completion of the EBG prediction (EBG + inference). We observed that over all 234 MSAs, EBG accounts for 19% and the ML inference for 81% of the total time-to-completion. SH-like aLRT and UFBoot2

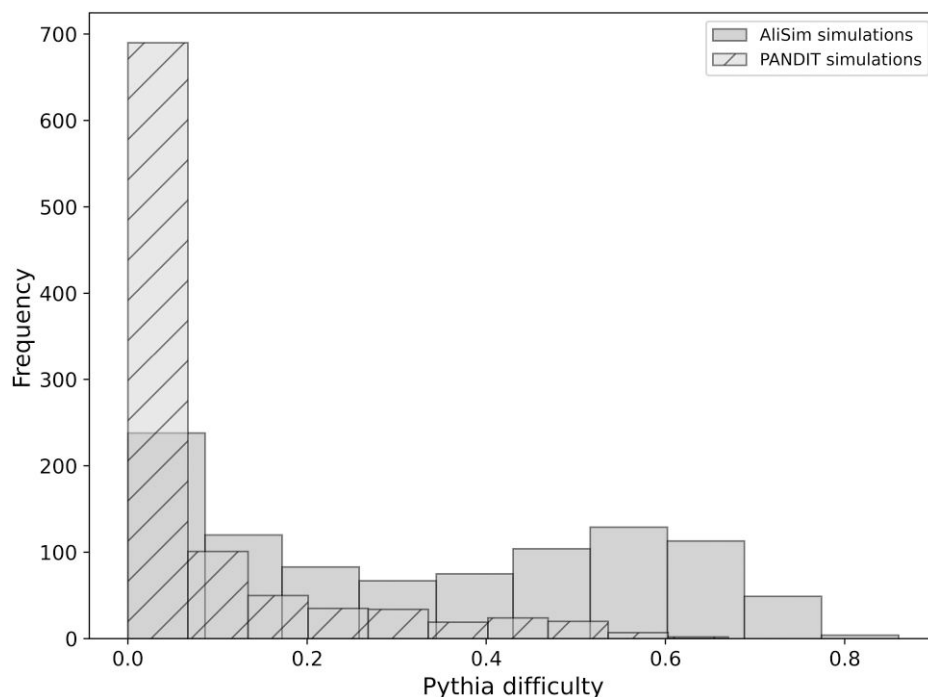


FIG. 6. Histogram of Pythia difficulties of the 979 AliSim (Ly-Trong et al. 2022) simulations used as well as 979 randomly sampled MSAs of the PANDIT (Whelan et al. 2003) DNA simulations from the original Ultrafast Bootstrap (Minh et al. 2013) evaluation.

Table 6. FP and true positive (TP) rates of the different support inference tools compared to the EBG classifier for a support threshold of 80

Tool	Filter value	Fraction considered	FP rate	TP rate
EBG	–	1.0	0.14	0.66
	0.80	0.89	0.10	0.68
	0.50	0.80	0.08	0.70
	0.30	0.73	0.06	0.71
	0.17	0.63	0.05	0.77
	0.16	0.48	0.00	0.00
SH-aLRT	N/A	1.0	0.17	0.74
UFBoot2	N/A	1.0	0.23	0.85
SBS	N/A	1.0	0.15	0.72
RB	N/A	1.0	0.15	0.72

The FP rate is the fraction of branches for which the support exceeds 80, but the respective branch is not in the true tree. The interpretation of TP is analogous. The fraction considered describes the fraction of analyzed branches after applying EBG’s uncertainty filtering with the respective uncertainty filter value.

exhibit a similar time-to-completion, both for DNA, and AA MSAs. For 97% of the datasets (DNA and AA), EBG outperformed UFBoot2 in terms of time-to-completion, with an average speedup of 9.4 ($\sigma = 5.5$). Considering MSAs of size $\geq 200,000$, EBG yielded an average speedup of 2.8 in comparison to UFBoot2. We observe an average speedup of 104 ($\sigma = 82$) of EBG compared to SBS when considering MSAs of size $\geq 200,000$. With increasing MSA sizes, the time-to-completion differences between UFBoot2 and EBG gradually decrease for AA datasets. We do not observe an analogous trend for DNA data.

Additionally, we used a random sample of 84 MSAs from the total of 234 MSAs to assess the disparity in accumulated CPU time between EBG and UFBoot2. In the median, running EBG along with an adaptive RAXML-NG ML tree search requires 28% less accumulated CPU time in

Table 7. Performance evaluation of the EBG regressor, EBG classifier with SBS threshold $t = 0.80$, and RB

Tool	MBE	MAE	MdAE	RMSE	BAC	F1	AUC
EBG	0.0	8.7	6.0	12.8	0.89	0.87	0.89
EBG*	0.1	7.1	4.0	11.1	0.95	0.94	0.95
RB	0.0	4.5	2.0	8.0	0.97	0.96	0.97

*EBG with uncertainty filtering

comparison to the corresponding UFBoot2 execution. However, summed over all 84 MSAs EBG including the inference, requires 57% more accumulated CPU time (65,137 vs. 43,090 s) than UFBoot2. This suggests that there are some outliers where UFBoot2’s time-to-completion is substantially smaller than EBG’s time-to-completion when we include the ML inference time in the calculation. According to our analysis, this occurs primarily on MSAs with a Pythia difficulty of ≥ 0.5 (see [supplementary section 8, Supplementary Material](#) online for a detailed visualization). Furthermore, we observe, that on the same 84 MSAs, EBG only accounts for 4% of the accumulated CPU time while the remaining 96% are required for the adaptive RAXML-NG tree inference that is needed as an input for EBG.

We conclude that it is challenging to devise a fair time-to-completion comparison between EBG and UFBoot2, as for EBG, we are undecided whether the ML inference time should be included or not. In contrast, for UFBoot2, it represents an intrinsic requirement for the computation of support values, as the ML search and the support calculations are necessarily intertwined. For instance, on 10 randomly selected datasets, the average time to completion of IQ-Tree increased by 22% with

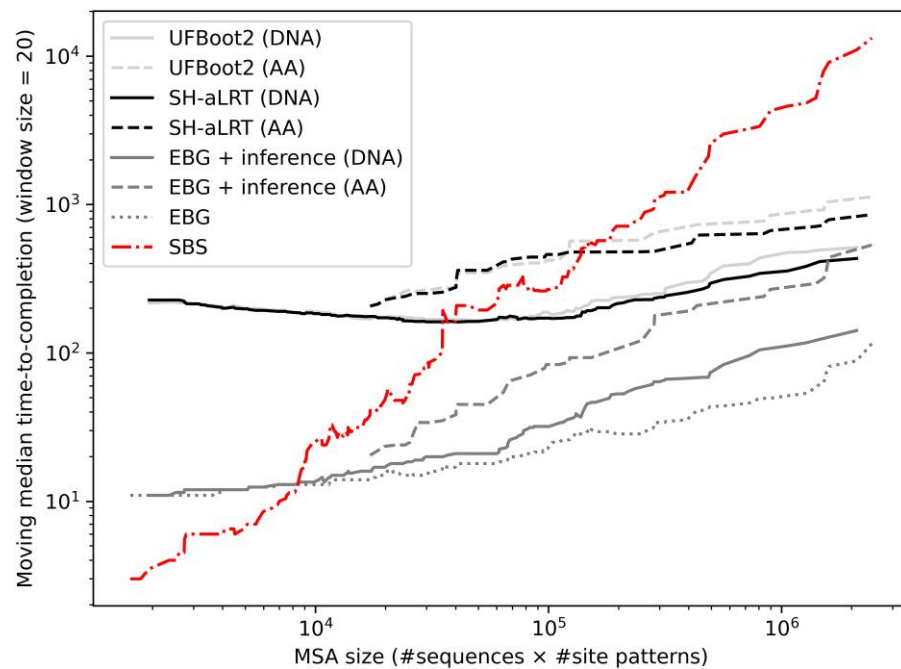


Fig. 7. Time-to-completion comparison for datasets of varying sizes with a moving average of window size 20.

UFBoot2 enabled. Hence, the above time comparisons reflect, to a large extent, a time comparison between the adaptive RAXML-NG and IQ-TREE2 ML search algorithms.

Performance on Edge Cases

To demonstrate the performance of EBG on an edge case, we predicted the SBS values for an MSA that was shown to be difficult to analyze in terms of phylogenetic inference. Morel et al. (2020) demonstrate the difficulties of obtaining a reliable ML phylogeny on a set of SARS-CoV-2 genome sequences. This difficulty is caused by the combination of a large number of sequences and a relatively low mutation rate, resulting in numerous branches with low bootstrap values. For our experiment, we used a set of 1,654 complete SARS-CoV-2 genomes, each with a length of 29,800 base pairs. We employed 100 RAXML-NG searches to determine the ML tree with the highest log-likelihood. We then performed an SBS run with 1,000 replicates to establish the ground truth SBS values. This analysis shows that only 13.9% of the inner branches of the ML tree yield an SBS value >70 , indicating an overall low support of the ML tree.

Predicting the branch support using our EBG regressor results in an overall good performance, with an MAE of 3, an MdAE of 0, and an RMSE of 9. Meanwhile, the EBG classifier, with an SBS threshold of $t := 70$, achieved a BAC of 0.82, an F1 of 0.77, and an AUC of 0.82. Therefore, even on an MSA that is known to be difficult-to-analyze, EBG can provide an accurate estimate of the SBS values for the corresponding ML tree. The prediction of the bootstrap support, using a mid-class laptop equipped with 4 cores and 8 GB of memory, has a time-to-completion of approximately 3 h. In contrast, computing the ground truth SBS values took 35 h, utilizing 10 nodes of a large computing cluster, each equipped with an Intel Xeon Gold 6230 (20 cores, 2.1

GHz) and 96 GB memory. On the same amount of computing nodes, the RB computation using the MPI version of RAXML required 5.5 h, whereas the MPI-based UFBoot2 analysis only took 31 min.

Another dataset that is notoriously difficult to analyze is the internal transcribed spacer (ITS) 354 (Grimm et al. 2007). ITS 354 is a short alignment (348 MSA sites) extracted from the ITS genes from 354 maple tree genomes. Predicting the SBS values using the EBG regressor results in an MAE of 6, an MdAE of 4, and an RMSE of 9. Hence, on this difficult MSA EBG also yields a good accuracy.

Finally, we tested EBG on the edge case of a deep time-scale phylogeny. We used an amino acid MSA of 11 insects, including damselfly *Calopteryx* (Ioannidis et al. 2017). On this dataset, EBG performs analogously to our systematic experiments with an MAE of 8.88.

Discussion and Conclusion

In this paper, we introduced EBG, a novel machine learning-based approach for predicting SBS values on phylogenies. For a given phylogeny and the respective MSA, EBG predicts point estimates of the SBS values for all inner branches, a probability of the SBS values exceeding a certain threshold, and uncertainty estimates for both prediction scenarios.

While EBG exhibits a high prediction accuracy, we do observe that RB remains the most accurate approximation method for directly predicting SBS values. Yet, we also demonstrate how the EBG uncertainty measures can help to reduce the accuracy gap to RB. By filtering the predictions of the EBG classifier to a filter value of $u \leq 0.7$, we can close this accuracy gap. EBG, even when including the ML tree inference time, requires substantially lower time-to-completion compared to the major competitor

UFBoot2 with an average speedup of 9.4 ($\sigma = 5.5$). This speedup comes at the cost of an increase in total accumulated CPU time summed over all test MSAs of 57% due to outliers.

On 979 simulated MSAs, EBG, UFBoot2, and SH-like aLRT are generally overconfident in predicting the true SBS values. Among the analyzed tools, EBG yields support predictions that are closest to the true branch support value.

EBG was trained and evaluated on phylogenies under the GTR+G model. Incorporating additional models and investigating EBG's performance on a variety of models is subject to future work.

Currently, EBG implements the sampling part of the PB procedure in Python. The PB is a performance bottleneck that substantially contributes to the prediction time and accounts for up to 70% of the overall time-to-completion of EBG excluding the ML inference time. Furthermore, EBG performs 204 RAXML-NG calls: one for each of the 200 parsimony bootstrap inferences, one for the parsimony inference, two for the support computation of both, and one for the nRF distance computation. Additionally, EBG stores and retrieves the results of those calls in individual files, inducing a potential I/O overhead. Unifying EBG's feature computation and the prediction as a command implemented within the RAXML-NG tool would likely result in a substantial speedup and streamline the entire prediction process. Since RAXML-NG is developed in our lab, the integration of EBG into RAXML-NG constitutes future work.

While EBG successfully establishes lower bounds for the EBG regression, we were not able to devise a prediction for the upper bound of SBS values. In our experiments, the attempts to optimize for any upper bound were unsuccessful, since they converged to the trivial upper bound of an SBS of 100. An upper bound for EBG regression would be useful to construct a comprehensive prediction interval. Future research could focus on the development of a method or model that is capable of reliably estimating upper bounds. A more informative prediction interval will be highly beneficial for assessing the range of potential SBS values. Future research might explore if EBG's uncertainty filtering can also enhance the performance of alternative support estimation tools such as SH-aLRT or UFBoot2. Since EBG is currently the only tool that can infer support uncertainty estimates, applying EBG-based filtering might be beneficial for other support inference tools as well.

Practitioners should be aware of the following recommendations when using EBG:

- The reference ML phylogeny should be inferred under the GTR+G model.
- In our experiments, the EBG classifier yields a more robust SBS prediction in the critical range ($60 \leq \text{SBS} \leq 95$) than the EBG regressor. Thus, we recommend to primarily use the classifier.
- We strongly recommend using EBG's uncertainty measures for filtering out uncertain predictions to

avoid incorrect conclusions. Table 6 and Fig. 3 shall serve as references for interpreting the uncertainty estimates of the EBG classifier.

We compare a comprehensive set of methods for SBS support estimation and approximation. EBG aims to rapidly and accurately predict the SBS values for a given phylogeny, in particular for the purpose of dataset exploration. However, depending on the biological question at hand, the anticipated downstream analyses, and if the goal is to obtain and extensively discuss publication-quality support values, we recommend to also calculate SBS values.

Finally, we note that parsimony-based methods may experience a renaissance, since as we show here and as already demonstrated by the Pythia tool, they constitute the by far most important and computationally inexpensive feature for conducting predictions about ML method results.

Supplementary Material

Supplementary material is available at *Molecular Biology and Evolution* online.

Funding

This work was financially supported by the Klaus Tschira Foundation and by the European Union (EU) under Grant Agreement No. 101087081 (Comp-Biodiv-GR).

Conflict of Interest. The authors declare that there are no conflicts of interest regarding the publication of this paper.

Data Availability

github.com/wiegertj/EBG github.com/wiegertj/EBG-train

References

- Ahmed SA, El-Mahallawy HS, Mohamed SF, Angelici MC, Hasapis K, Saber T, Karanis P. Subtypes and phylogenetic analysis of *Blastocystis* sp. isolates from West Ismailia, Egypt. *Sci Rep*. 2022;**12**(1):19084. <https://doi.org/10.1038/s41598-022-23360-0>.
- Akiba T, Sano S, Yanase T, Ohta T, Koyama M. Optuna: a next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19; 2019, Association for Computing Machinery. p. 2623–2631. <https://doi.org/10.1145/3292500.3330701>.
- Anisimova M, Gascuel O. Approximate likelihood-ratio test for branches: a fast, accurate, and powerful alternative. *Syst Biol*. 2006;**55**(4):539–552. <https://doi.org/10.1080/10635150600755453>.
- Bavelas A. Communication patterns in task-oriented groups. *J Acoust Soc Am*. 2005;**22**(6):725–730. <https://doi.org/10.1121/1.1906679>.
- Botchkarev A. A new typology design of performance metrics to measure errors in machine learning regression algorithms. *Int J Inf Knowl Manage*. 2019;**14**:045–076. <https://doi.org/10.28945/4184>.
- Brandis G. Reconstructing the evolutionary history of a highly conserved operon cluster in gammaproteobacteria and bacilli.

- Genome Biol Evol. 2021;**13**(4):evab041. <https://doi.org/10.1093/gbe/evab041>.
- Buckley TR, Cunningham CW. The effects of nucleotide substitution model assumptions on estimates of nonparametric bootstrap support. *Mol Biol Evol.* 2002;**19**(4):394–405. <https://doi.org/10.1093/oxfordjournals.molbev.a004094>.
- Cruaud A, Delvare G, Nidelet S, Sauné L, Ratnasingham S, Chartois M, Blaimer BB, Gates M, Brady SG, Faure S, et al. Ultra-conserved elements and morphology reciprocally illuminate conflicting phylogenetic hypotheses in Chalcididae (Hymenoptera, Chalcidoidea). *Cladistics.* 2021;**37**(1):1–35. <https://doi.org/10.1111/cla.v37.1>.
- Dunlap WP, Corey DM, Burke MJ. Averaging correlations: expected values and bias in combined Pearson r s and Fisher's z transformations. *J Gen Psychol.* 1998;**125**(3):245–261. <https://doi.org/10.1080/00221309809595548>.
- Felsenstein J. Confidence limits on phylogenies: an approach using the bootstrap. *Evolution.* 1985;**39**(4):783–791. <https://doi.org/10.2307/2408678>.
- Felsenstein J, Kishino H. Is there something wrong with the bootstrap on phylogenies? A reply to Hillis and bull. *Syst Biol.* 1993;**42**(2):193–200. <https://doi.org/10.1093/sysbio/42.2.193>.
- Grimm GW, Renner SS, Stamatakis A, Hemleben V. A nuclear ribosomal DNA phylogeny of *Acer* inferred with maximum likelihood, splits graphs, and motif analysis of 606 sequences. *Evol Bioinform Online.* 2007;**2**:7–22. <https://doi.org/10.1177/117693430600200014>.
- Guindon S, Dufayard J-F, Lefort V, Anisimova M, Hordijk W, Gascuel O. New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0. *Syst Biol.* 2010;**59**(3):307–321. <https://doi.org/10.1093/sysbio/syq010>.
- Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. *Mach Learn.* 2002;**46**(1):389–422. <https://doi.org/10.1023/A:1012487302797>.
- Haag J, Höhler D, Bettisworth B, Stamatakis A. From easy to hopeless—predicting the difficulty of phylogenetic analyses. *Mol Biol Evol.* 2022;**39**(12):msac254. <https://doi.org/10.1093/molbev/msac254>.
- Hawkins DM. The problem of overfitting. *J Chem Inf Comput Sci.* 2004;**44**(1):1–12. <https://doi.org/10.1021/ci0342472>.
- Hoang DT, Chernomor O, von Haeseler A, Minh BQ, Vinh LS. UFBoot2: improving the ultrafast bootstrap approximation. *Mol Biol Evol.* 2018;**35**(2):518–522. <https://doi.org/10.1093/molbev/msx281>.
- Huelsenbeck JP, Hillis DM. Success of phylogenetic methods in the four-taxon case. *Syst Biol.* 1993;**42**(3):247–264. <https://doi.org/10.1093/sysbio/42.3.247>.
- Ioannidis P, Simao FA, Waterhouse RM, Manni M, Seppely M, Robertson HM, Misof B, Niehuis O, Zdobnov EM. Genomic features of the damselfly calopteryx splendens representing a sister clade to most insect orders. *Genome Biol Evol.* 2017;**9**(2):415–430. <https://doi.org/10.1093/gbe/evx006>.
- Kapli P, Yang Z, Telford MJ. Phylogenetic tree building in the genomic age. *Nat Rev Genet.* 2020;**21**(7):428–444. <https://doi.org/10.1038/s41576-020-0233-0>.
- Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T-Y. Lightgbm: a highly efficient gradient boosting decision tree. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17*; 2017. Curran Associates Inc. p. 3149–3157. <https://doi.org/10.5555/3294996.3295074>.
- Koenen EJM, Ojeda DI, Steeves R, Migliore J, Bakker FT, Wieringa JJ, Kidner C, Hardy OJ, Pennington RT, Bruneau A, et al. Large-scale genomic sequence data resolve the deepest divergences in the legume phylogeny and support a near-simultaneous evolutionary origin of all six subfamilies. *New Phytol.* 2020;**225**(3):1355–1369. <https://doi.org/10.1111/nph.v225.3>.
- Koenker R, Bassett G. Regression quantiles. *Econometrica.* 1978;**46**(1):33–50. <https://doi.org/10.2307/1913643>.
- Kozlov AM, Darriba D, Flouri T, Morel B, Stamatakis A. Raxml-ng: a fast, scalable and user-friendly tool for maximum likelihood phylogenetic inference. *Bioinformatics.* 2019;**35**(21):4453–4455. <https://doi.org/10.1093/bioinformatics/btz305>.
- Leuchtenberger AF, Crotty SM, Drucks T, Schmidt HA, Burgstaller-Muehlbacher S, von Haeseler A. Distinguishing Felsenstein zone from Farris zone using neural networks. *Mol Biol Evol.* 2020;**37**(12):3632–3641. <https://doi.org/10.1093/molbev/msaa164>.
- Ly-Trong N, Naser-Khdour S, Lanfear R, Minh BQ. AliSim: a fast and versatile phylogenetic sequence simulator for the genomic era. *Mol Biol Evol.* 2022;**39**(5):msac092. <https://doi.org/10.1093/molbev/msac092>.
- Minh BQ, Nguyen MAT, von Haeseler A. Ultrafast approximation for phylogenetic bootstrap. *Mol Biol Evol.* 2013;**30**(5):1188–1195. <https://doi.org/10.1093/molbev/mst024>.
- Morel B, Barbera P, Czech L, Bettisworth B, Hübner L, Lutteropp S, Sordani D, Kostaki E-G, Mamami I, Kozlov AM, et al. Phylogenetic analysis of SARS-CoV-2 data is difficult. *Mol Biol Evol.* 2020;**38**(5):1777–1791. <https://doi.org/10.1093/molbev/msaa314>.
- Pattengale ND, Alipour M, Bininda-Emonds OR, Moret BM, Stamatakis A. How many bootstrap replicates are necessary? *J Comput Biol.* 2010;**17**(3):337–354. <https://doi.org/10.1089/cmb.2009.0179>.
- Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, et al. Scikit-learn: machine learning in python. *J Mach Learn Res.* 2011;**12**(85):2825–2830. <https://doi.org/10.5555/1953048.2078195>.
- Piel WH, Donoghue MJ, Sanderson MJ. Treebase v. 2: a database of phylogenetic knowledge. *e-BioSphere* 2009; 2009.
- Robinson DF, Foulds LR. Comparison of phylogenetic trees. *Math Biosci.* 1981;**53**(1–2):131–147. [https://doi.org/10.1016/0025-5564\(81\)90043-2](https://doi.org/10.1016/0025-5564(81)90043-2).
- Shannon CE. A mathematical theory of communication. *Bell Syst Tech J.* 1948;**27**(3):379–423. <https://doi.org/10.1002/bltj.1948.27.issue-3>.
- Stamatakis A. RAXML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics.* 2014;**30**(9):1312–1313. <https://doi.org/10.1093/bioinformatics/btu033>.
- Stamatakis A, Hoover P, Rougemont J. A rapid bootstrap algorithm for the RAXML web servers. *Syst Biol.* 2008;**57**(5):758–771. <https://doi.org/10.1080/10635150802429642>.
- Steck M, Porter ML, Lutz H, Chong RA. The complete mitochondrial genomes and phylogenetic analysis of two Nycteribiidae bat flies (Diptera: hippoboscoidea). *Mitochondrial DNA Part B.* 2022;**7**(8):1486–1488. <https://doi.org/10.1080/23802359.2022.2107450>.
- Susko E. Bootstrap support is not first-order correct. *Syst Biol.* 2009;**58**(2):211–223. <https://doi.org/10.1093/sysbio/syp016>.
- Togkousidis A, Kozlov OM, Haag J, Höhler D, Stamatakis A. Adaptive RAXML-NG: accelerating phylogenetic inference under maximum likelihood using dataset difficulty. *Mol Biol Evol.* 2023;**40**(10):msad227. <https://doi.org/10.1093/molbev/msad227>.
- Trost J, Haag J, Höhler D, Nesterenko L, Jacob L, Stamatakis A, Boussau B. Simulations of sequence evolution: how (un)realistic they really are and why. *bioRxiv* 548509. <https://doi.org/10.1101/2023.07.11.548509>, 12 July 2023, preprint: not peer reviewed.
- Whelan S, de Bakker PIW, Goldman N. Pandit: a database of protein and associated nucleotide domains with inferred trees. *Bioinformatics.* 2003;**19**(12):1556–1563. <https://doi.org/10.1093/bioinformatics/btg188>.