OPT2022: 14th Annual Workshop on Optimization for Machine Learning

Completing the Model Optimization Process by Correcting Patterns of Failure in Regression Tasks

Thomas Bonnier THOMAS.BONNIER@CENTRALIENS-LILLE.ORG
Centrale Lille Alumni

Abstract

Model selection and hyper-parameter optimization sometimes prove to be complex and costly processes with unfinished outcomes. In fact, a so-called optimized model can still suffer from patterns of failure when predicting on new data, affecting the generalization error. In this paper, we focus on regression tasks and introduce an additional stage to the model optimization process in order to render it more reliable. This new step aims to correct error patterns when the model makes predictions on unlabeled data. To that end, our method includes two techniques. *AutoCorrect Rules* leverage the model under/overestimation bias and applies simple rules to adjust predictions. *AutoCorrect Model* is a supervised approach which exploits different representations to predict residuals in order to revise model predictions. We empirically prove the relevance of our method on the outcome of an AutoML tool using different time budgets, and on a specific optimization case leveraging a pre-trained model for an image regression task.

1. Introduction

As the learner selection and hyper-parameter tuning process can prove to be complex, various Automated Machine Learning (AutoML) libraries have been proposed to address this task for a given training dataset and optimization metric. Those tools have specificities with different search strategies, such as Bayesian optimization with meta-learning and model ensembles [12], combining the strengths of Bayesian optimization and Hyperband [11], or ECI-based (Estimated Cost for Improvement) learner choice and cost-effective hyper-parameter tuning [29]. Still, Machine Learning (ML) models are inclined to failure modes when error patterns surface during inference [28]. Even though a model displays good overall performance, some regions of the feature space may be prone to higher error rates. For example, a regression model could be subject to an overestimation bias in a given area of the feature space. These failure patterns turn out to be harmful in regression applications such as price prediction in finance or dysmorphic facial sign detection in the medical field.

Scope and contribution In this paper, the focus is on regression tasks. We consider the output of an optimization process, i.e. an ML model M^*, based on an initial configuration $\chi = (\mathcal{M}, \mathcal{H}, D^t, T, m, e)$, where $\mathcal{M}$ and $\mathcal{H}$ denote the algorithm space and the hyper-parameter space, respectively. D^t is the training dataset. T is the time budget. m denotes an objective metric to be minimized (e.g. mean squared error MSE) and e is an evaluation method (e.g. cross-validation). M^* has been trained on $D^t(\boldsymbol{X}, Y)$, with the features $\boldsymbol{X}$ and target variable Y, and is ready to be deployed in order to make predictions on new (unlabeled) data. We aim to extend the model optimization process with an additional stage in order to give unbiased and reliable generalization properties to M^*. This new step is intended for correcting potential patterns of failure surfacing during inference. We do not alter the parameters of M^*, but we adjust its predictions based on error patterns detected by our method. To that end, we propose two techniques. In order to assess the ML model under/overestimation bias, **AutoCorrect Rules** first calibrates rules based on the model prediction values on a dataset with residuals ($Y - \hat{Y}$). The prediction is then adjusted during inference to counteract any potential error bias. **AutoCorrect Model** leverages $\boldsymbol{X}$ and prediction $\hat{Y} = M^*(\boldsymbol{X})$ as features, and the residuals as target variable. When the ML model predicts based on new data, AutoCorrect Model then estimates the residual values in order to adjust the original ML model predictions. We experimentally show the relevance of those techniques on two types of use cases: (i) On tabular data where the ML model has been optimized via an AutoML tool with various time budgets. (ii) On image data where the ML model includes a frozen pre-trained model and additional top layers tuned on new data. This second application is thus a specific optimization use case based on transfer learning.

2. AutoCorrect method

Notations and assumptions We consider a training dataset D^t with $\boldsymbol{x}_i, y_i \in \mathcal{X} \times \mathcal{Y}$ for $i = 1, ..., N$ of i.i.d. realizations of random variables $\boldsymbol{X}, Y \sim \mathbb{P}$, where the joint distribution $\mathbb{P}$ is unknown. The ML model is a predictor $M^* : \mathcal{X} \to \mathcal{Y}$ with $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}$. M^* is the outcome of an optimization process (e.g. AutoML): $argmin_{M_h, \boldsymbol{h}}(m(M_h, D^t))$, subject to budget T and evaluation method e, and where algorithm $M_h \in \mathcal{M}$ and hyper-parameters $\boldsymbol{h} \in \mathcal{H}$, as defined in the previous section. At the time of inference, the evaluation set $\{\boldsymbol{x}_i\}_{i>N}$ is unlabeled.

Prerequisite AutoCorrect Rules and Model both require a calibration dataset including instances $\boldsymbol{x}_i, y_i$, the corresponding ML model predictions $\hat{y}_i$, and thus the residuals $r_i = y_i - \hat{y}_i$. These should be computed on data which has not been not used to train M^*. Similarly to cross-validation, we randomly partition D^t into k mutually exclusive folds $D^t_1, ..., D^t_k$ of equal size. M^* is repeatedly trained k times on $D^t \backslash D^t_j$, with $j \in 1, ..., k$. The model predictions, and thus the residuals, are computed based on D^t_j. We thus obtain D^{t+}, the whole dataset with residuals. Lastly, we train M^* on the whole dataset D^t. We note that this process can be naturally included in a model selection and hyper-parameter optimization process (i.e. AutoML tool). According to Kohavi [18], if the ML model remains stable under the perturbations created by removing each fold, the cross-validation estimate of the generalization error will remain unbiased. Better stability should be observed with a large number of subsets. Further, for most models, the variance of the estimate of the error rate will either be constant or decreasing with k. We follow the author's suggestion to use ten folds as a reasonable trade-off between stability and computational cost.

2.1. AutoCorrect Rules

The residual plot, showing the residuals r_i against the fitted values $\hat{y}_i$, is a diagnostic graph employed to check whether the variance of residuals is constant in linear regression models [1]. We exploit this representation and apply it to ML models, in order to assess any potential bias in the residuals, based on the ML model prediction values. $r_i > 0$ amounts to a model underestimation bias as $y_i > \hat{y}_i$. To this end, we use the dataset with residuals D^{t+} and define P^* interval bins B_p. Each B_p is the set of samples whose prediction values $\hat{y}$ fall into the interval I_p. I_p are equally-spaced intervals of $\hat{y}$: $I_p = [m^- + (m^+ - m^-) \times \frac{p-1}{P^*}, m^- + (m^+ - m^-) \times \frac{p}{P^*})$, where $m^- = min(\hat{y}_i)$ and $m^+ = max(\hat{y}_i)$, with $1 \leq p \leq P^*$ and $1 \leq i \leq N$. We com-

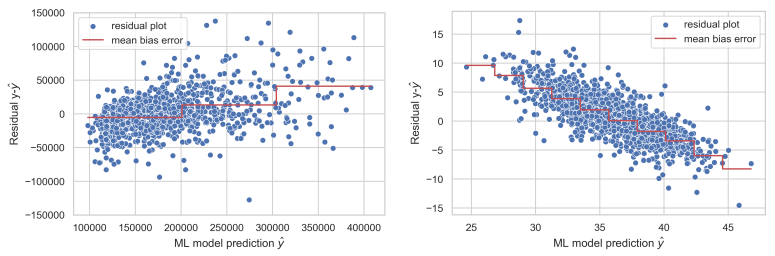

Figure 1: Residual plots $y - \hat{y}$ against $\hat{y}$, with slight (left: use case 1 from experiments) and severe (right: use case 3) residual patterns. The residual plot (blue dots) shows patterns based on the ML model predicted value. The mean bias error (MBE) is displayed on equally-spaced intervals of $\hat{y}$ on the dataset with residuals (red line). Right plot: there is a clear underestimation bias for small values of $\hat{y}$ and an overestimation bias for large values.

pute the mean bias error $MBE(B_p) = \frac{1}{N_p}\sum_{j=1}^{N_p} r_j$ for each B_p, where N_p denotes the number of samples in B_p. Figure 1 shows two types of residual plots with different degrees of error pattern. During inference on a new sample $\boldsymbol{x'}$, the prediction is adjusted based on $M^*(\boldsymbol{x'})$ value: $\hat{y}'_{adj} = M^*(\boldsymbol{x'}) + MBE(B_p)_{M^*(\boldsymbol{x'})\in I_p}$ (with potential boundaries on $\hat{y}'_{adj}$ depending on the use case, e.g. positive amounts). If $M^*(\boldsymbol{x'}) < min(\hat{y}_i)$ or $M^*(\boldsymbol{x'}) > max(\hat{y}_i)$, then the applied correction is $MBE(B_1)$ or $MBE(B_{P^*})$, respectively. Lastly, the optimal number of bins is computed with D^{t+} using cross-validation.

2.2. AutoCorrect Model

While rules can be relevant for some use cases, predicting residual values with a regression model may prove to be more effective. D^{t+} is thus used to train a new regression model $A^* : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, with $\mathcal{X} \subseteq \mathbb{R}^d$, $\mathcal{Y} \subseteq \mathbb{R}$ and $\mathcal{Z} \subseteq \mathbb{R}$: $A^* = argmin_A(\frac{1}{N}\sum_{i=1}^{N} L(r_i, A(\boldsymbol{x}_i, \hat{y}_i)))$. A is chosen in $\mathcal{M}$, and L is a loss function consistent with the objective metric m: for example, if m is the mean squared error, L will be the squared loss. $A^*(\boldsymbol{X}, \hat{Y})$ is an estimate of $\mathbb{E}(R|\boldsymbol{X}, \hat{Y})$, with $R = Y - \hat{Y}$. During inference on a new sample $\boldsymbol{x'}$, the prediction is adjusted: $\hat{y}'_{adj} = M^*(\boldsymbol{x'}) + A^*(\boldsymbol{x'}, \hat{y}')$ (with potential boundaries on $\hat{y}'_{adj}$ depending on the use case), with $\hat{y}' = M^*(\boldsymbol{x'})$. A^* can leverage various representations as features, such as $\boldsymbol{X}$ and/or $\hat{Y}$. When $\boldsymbol{X}$ is sparse or if $d >> N$, more parsimonious representations can be adequate. For instance, in a multi-target regression setting where $\mathcal{Y} \subseteq \mathbb{R}^k$ with $k > 1$, any dependency between the targets could be exploited. $A^*(\hat{\boldsymbol{Y}})$ may then be helpful to learn error patterns and correct the predictions at inference time. In the multi-target case, A^* could be either a specialized model correcting predictions related to one target Y^j (i.e. r^j is the target used to train A^*) or a general model with several targets $\boldsymbol{Y}$. Algorithm 1 gives an outline of the whole AutoCorrect learning approach.

Algorithm 1: AutoCorrect

```
Input: Training data $D^t$; ML model $M^*$;
        Number of folds: $K$;
for $j = 1, ..., K$ do
    Train $M^*$ on $D^t \backslash D^t_j$;
    Predict $\hat{y}_i$ with $M^*$ on $D^t_j$;
    Compute $r_i = y_i - \hat{y}_i$ on $D^t_j$;
    $D^{t+}_j = D^t_j \cup \{\hat{y}_i, r_i\}_{i \in D^t_j}$;
end
$D^{t+} = \bigcup (D^{t+}_j)_{1 \leq j \leq K}$;
```

Algorithm 1 (continued): AutoCorrect

```
Train $M^*$ on full $D^t$;
Optimize number of bins $P^*$ using $D^{t+}$;
Compute interval bins $B_p$ $(1 \leq p \leq P^*)$;
/* AutoCorrect Rules and Model */;
for $j = 1, ..., P^*$ do
    $MBE(B_p) = \frac{1}{N_p} \sum_{j=1}^{N_p} r_j$;
end
$A^* = argmin_A(\frac{1}{N} \sum_{i=1}^{N} L(r_i, A(\boldsymbol{x}_i, \hat{y}_i)))$;
Output: $MBE(B_p)_{1 \leq p \leq P^*}$; $A^*$
```

3. Experiments

3.1. Assessing the outcome of an AutoML tool

Datasets With the Ames Housing Dataset (use case 1), we aim to predict the sale price of residential properties in Iowa, with 80 explanatory variables [6, 7]. We use the 1,460 instances for which the target is available. The Seoul Bike Sharing Demand Dataset (use case 2) is a tabular dataset with 8,760 instances. The target is the count of public bikes rented every hour in Seoul [9, 10, 24]. In each case, 20% of the original dataset is kept for testing and the rest is used as D^t.

Modeling settings The dataset with residuals is constructed with a 10-fold strategy. The number of bins in AutoCorrect Rules is optimized using 5-fold cross-validation with values in $[2, 10]$. Experiments are run over 4 seeds (distinct dataset splits). M^* and A^* are both selected and optimized using Fast Lightweight AutoML (FLAML) [22, 29] with various time budgets, and with the MSE optimization metric. Time budget includes learner selection and hyper-parameter tuning. A^* budget is set to $\min(M^* \text{ budget}, 300s)$. Experiments are run on an Intel i7-7500U CPU with 12 GB RAM. Learners are selected among Random Forest [3], LGBM [16], XGBoost [5], CatBoost [23], and Extra Trees [13].

FLAML includes a specific cost-related algorithm for tuning hyper-parameters [30], called Cost-Frugal Optimization (CFO) method. This randomized search algorithm starts in a low-cost region, moves gradually towards a low-loss area, and runs until the end of the budget. $\boldsymbol{h}$ denotes the hyper-parameter configuration, f and g are the respective loss and cost function, and δ refers to the step size. The initial point $\boldsymbol{h}_0$ has a small $g(\boldsymbol{h}_0)$. At each iteration, a vector $\boldsymbol{u}$ is sampled uniformly at random from the unit sphere:

- if $f(\boldsymbol{h} + \delta \boldsymbol{u}) < f(\boldsymbol{h})$, then $\boldsymbol{h} \leftarrow \boldsymbol{h} + \delta \boldsymbol{u}$,
- else if $f(\boldsymbol{h} - \delta \boldsymbol{u}) < f(\boldsymbol{h})$, then $\boldsymbol{h} \leftarrow \boldsymbol{h} - \delta \boldsymbol{u}$.

If no progress is being made or in order to avoid getting confined in a local optimum, δ can be dynamically adjusted or the process can restart from a randomized starting point. This approach provides guarantees for a convergence rate of $O(\sqrt{d}/\sqrt{K})$, with d denoting the search space dimensionality and K the number of iterations, and for an upper bounded cost.

Lastly, we also use A^*_{RF}, a simple Autocorrect model based on a random forest with 100 trees, without optimizing its hyper-parameters.

Table 1: RMSE between true values y and predictions on an independent test dataset (20% of initial dataset), before and after applying AutoCorrect, averaged over 4 seeds. When AutoCorrect Rules and Models ($A^*(\boldsymbol{x}, \hat{y})$, and $A^*_{RF}(\boldsymbol{x}, \hat{y})$) are used, we also display the relative change (%) versus M^* RMSE. RMSE increase is in red, best results are in bold.

Use case	M^* budget (s)	M^*	AutoCorrect Rules	With $A^*(\boldsymbol{x}, \hat{y})$	With $A^*_{RF}(\boldsymbol{x}, \hat{y})$
1	60	30,369.3	29,218.0 (-3.8)	27,502.1 (-9.4)	**25,789.7 (-15.1)**
	120	30,264.7	28,780.4 (-4.9)	26,464.5 (-12.6)	**26,458.5 (-12.6)**
	360	29,605.5	28,836.2 (-2.6)	27,408.5 (-7.4)	**25,715.0 (-13.1)**
	1,200	25,208.2	25,175.4 (-0.1)	24,633.8 (-2.3)	**24,282.7 (-3.7)**
	1,500	25,121.8	25,093.5 (-0.1)	24,074.2 (-4.2)	**23,722.5 (-5.6)**
2	60	200.9	199.6 (-0.6)	188.9 (-6.0)	**183.6 (-8.6)**
	120	185.1	184.9 (-0.1)	**178.4 (-3.7)**	179.8 (-2.9)
	360	171.0	170.7 (-0.2)	**168.1 (-1.7)**	169.7 (-0.7)
	1,200	168.3	168.3 (0.0)	**168.0 (-0.2)**	169.0 (0.4)
	1,500	168.3	168.3 (0.0)	**167.9 (-0.2)**	168.6 (0.2)

Results M^* is considered as the baseline. Table 1 shows the evolution of the test root mean squared error (RMSE) after applying AutoCorrect Rules and 2 different AutoCorrect Models: $A^*(\boldsymbol{x}, \hat{y})$ and $A^*_{RF}(\boldsymbol{x}, \hat{y})$. The relative change in error is, on average, almost always negative (a few minor exceptions in red), which demonstrates a reliable behavior. In use case 1, although M^* RMSE tends to converge with increasing time budget, AutoCorrect Models reveal that there is still potential for further optimization. On the other hand, in use case 2, with budgets greater than 360s, this potential proves to be thinner. In that case, the AutoML tool seems to have produced an adequate outcome. Lastly, $A^*_{RF}(\boldsymbol{x}, \hat{y})$ mean training time is 4.8s (case 1) and 8.3s (case 2). This simple model proves to be a useful low-cost alternative.

3.2. Optimizing a pre-trained model for image regression

Dataset The Facial Key Points Detection (use case 3) is a face image dataset based on 96×96 pixels [2]. The objective is to predict the (x,y) coordinates for 15 key points (e.g. nose tip), for applications such as dysmorphic facial sign detection. There are thus 30 target variables. We retain 2,140 entries, i.e. images without missing target values. 20% of the original dataset is kept for testing and the rest is used as D^t.

Table 2: RMSE results on an independent test dataset, before and after applying AutoCorrect, averaged over 4 seeds. A^* budget is set to 60s. For AutoCorrect Rules and Model $A^*(\hat{\boldsymbol{y}})$, we display the relative change (%) versus M^* RMSE. Best results are in bold.

Target variable	M^*	AutoCorrect Rules	With $A^*(\hat{\boldsymbol{y}})$	With $A^*_{RF}(\hat{\boldsymbol{y}})$
left eye center x	5.21	2.26 (-56.5)	**1.92 (-62.9)**	1.94 (-62.8)
left eye center y	3.36	2.29 (-31.8)	**2.05 (-39.3)**	2.06 (-38.6)

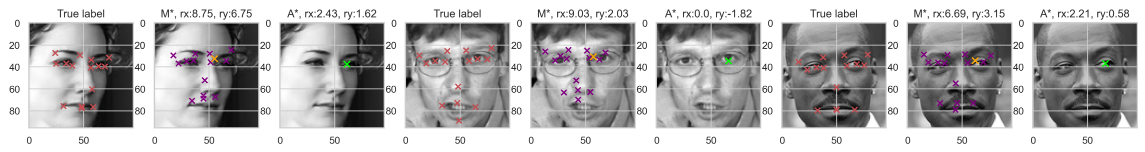

Figure 2: Each of these 3 examples from the test dataset shows: the true coordinates for the 15 key points (left), M^* predictions with an orange marker for the left eye center (middle), and the predictions for the left eye center coordinates adjusted by A^* (right). True residuals are indicated for each coordinate related to the left eye center: rx and ry.

Modeling settings M^* is trained on all the key points. It has the following architecture: pre-trained MobileNet v2 [26] with non-trainable weights, completed with global max pooling, a dense layer (128 units), batch normalization, ReLu activation, and a final layer for the 30 outputs. The MSE is minimized with Adam [17]. The number of epochs is tuned with an early stopping strategy (patience of 5) based on validation loss. We use AutoCorrect to optimize predictions on two target values: (x,y) for the left eye center. $A^*(\hat{\boldsymbol{y}})$ leverages the dependencies between the 30 coordinates to predict the residuals. We use one specific AutoCorrect Model ($\mathcal{Y} \rightarrow \mathcal{Z}$ with $\mathcal{Y} \subseteq \mathbb{R}^{30}$ and $\mathcal{Z} \subseteq \mathbb{R}$) for each coordinate of the left eye center in order to predict the corresponding residuals. Each A^* is optimized using FLAML, with $T = 60s$ and $m = MSE$.

Lastly, we also use $A^*_{RF}(\hat{\boldsymbol{y}})$, a simple Autocorrect model using a random forest with 100 trees, without hyper-parameter optimization.

Results Table 2 demonstrates the efficacy of AutoCorrect Rules and especially AutoCorrect Model. There are significant patterns in the residuals that are well exploited by both approaches. Figure 2 exhibits the effect of correcting with A^* on 3 test instances. Lastly, A^*_{RF} mean training time is 2.9s (left eye center x) and 3.0s (left eye center y). This model turns out to be a valuable low-cost alternative to A^*.

4. Conclusion and future work

In this paper, we presented AutoCorrect which aims to complete the optimization process and render it more reliable for a moderate additional time budget. On average, AutoCorrect in fact improves the generalization properties of a model by detecting patterns in the residuals. AutoCorrect could also be used in validation during the optimization process to assess whether an ML model requires further optimization. Moreover, additional AutoML tools could be tested and compared for given time budgets. More runs could be performed to better assess the reliability of the methodology. The practicality of AutoCorrect should also be assessed when the data size is large. Extending the optimization pipeline obviously adds complexity and computational costs. On the other hand, AutoCorrect is also a promise for potential improvement in performance and additional transparency. In fact, AutoCorrect can reveal patterns in the residuals whose explainability could be performed with methods such as Shapley values [21, 27]. Lastly, we could study the efficacy of AutoCorrect when distribution shifts occur.

References

[1] Francis J Anscombe. Graphs in statistical analysis. *The american statistician*, 27(1):17–21, 1973.

[2] Yoshua Bengio. Facial keypoints detection. https://www.kaggle.com/competitions/facial-keypoints-detection/data, 2016. Accessed: 2022-07-01.

[3] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[4] Jiefeng Chen, Frederick Liu, Besim Avci, Xi Wu, Yingyu Liang, and Somesh Jha. Detecting errors and estimating accuracy on unlabeled data with self-training ensembles. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 14980–14992, 2021.

[5] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM, 2016.

[6] Dean De Cock. House prices - advanced regression techniques. https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data, 2011. Accessed: 2022-07-01.

[7] Dean De Cock. Ames, iowa: Alternative to the boston housing data as an end of semester regression project. *Journal of Statistics Education*, 19(3), 2011.

[8] Pinar Donmez, Guy Lebanon, and Krishnakumar Balasubramanian. Unsupervised supervised learning I: estimating classification and regression errors without labels. *Journal of Machine Learning Research*, 11:1323–1351, 2010.

[9] Sathishkumar V. E. and Yongyun Cho. A rule-based model for seoul bike sharing demand prediction using weather data. *European Journal of Remote Sensing*, 53(sup1):166–183, 2020.

[10] Sathishkumar V. E., Jangwoo Park, and Yongyun Cho. Using data mining techniques for bike sharing demand prediction in metropolitan city. *Computer Communications*, 153:353–366, 2020.

[11] Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: robust and efficient hyperparameter optimization at scale. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1436–1445. PMLR, 2018.

[12] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2962–2970, 2015.

[13] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

[14] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 2017.

[15] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[16] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3146–3154, 2017.

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[18] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes*, pages 1137–1145, 1995.

[19] Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2801–2809. PMLR, 2018.

[20] Wen-Shing Lee, Doris L Grosh, Frank A Tillman, and Chang H Lie. Fault tree analysis, methods, and applications, a review. *IEEE Transactions on Reliability*, 34(3):194–203, 1985.

[21] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4765–4774, 2017.

[22] Microsoft. Flaml. https://github.com/microsoft/FLAML, 2021. Accessed: 2022-07-01.

[23] Liudmila Ostroumova Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6639–6649, 2018.

[24] UCI ML Repository. Seoul bike sharing demand data set. https://archive.ics.uci.edu/ml/datasets/Seoul+Bike+Sharing+Demand, 2020. Accessed: 2022-07-01.

[25] Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[26] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520, 2018.

[27] Lloyd S. Shapley. A value for n-person games. *Contributions to the Theory of Games II, Princeton University Press, Princeton*, pages 307–317, 1953.

[28] Sahil Singla, Besmira Nushi, Shital Shah, Ece Kamar, and Eric Horvitz. Understanding failures of deep networks via robust feature extraction. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 12853–12862. Computer Vision Foundation / IEEE, 2021.

[29] Chi Wang, Qingyun Wu, Markus Weimer, and Erkang Zhu. FLAML: A fast and lightweight automl library. In *Proceedings of Machine Learning and Systems 2021, MLSys 2021, virtual, April 5-9, 2021*. mlsys.org, 2021.

[30] Qingyun Wu, Chi Wang, and Silu Huang. Frugal optimization for cost-related hyperparameters. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 10347–10354. AAAI Press, 2021.

Appendix A. Related work

The focus of research related to error detection and error rate estimation has mainly been on neural networks and classification tasks. Singla et al. [28] adapted the Fault Tree Analysis [20] to deep neural networks by representing the model failure modes with interpretable features in classification tasks.

Leveraging the model confidence, such as the maximum softmax probability in classification tasks, is a first step to detecting incorrect predictions during inference [15]. However, modern neural networks have proved to be poorly calibrated [14]. Kuleshov et al. [19] extend calibration to regression in order to produce uncertainty estimates. According to them, a forecaster M is calibrated if the empirical cumulative function (CDF), based on the target values y_t, matches the predicted CDF, based on the outputs of M, when the dataset size goes to infinity.

The problem of accuracy estimation with unlabeled data has received more attention on classification than on regression tasks. For instance, Chen et al. [4] employ an ensemble of models to detect errors and estimate a classifier's accuracy. With regard to regression, Donmez et al. [8] estimate the error rate assuming that the target distribution $p(y)$ is known, in order to evaluate the maximum likelihood of the predictor outputs.

With regard to model uncertainty in regression tasks, Romano et al. suggest combining the strengths of conformal prediction and quantile regression [25]. This method, called *conformalized quantile regression*, produces valid and shorter prediction intervals, with lengths varying according to heteroskedasticity in the data.

Appendix B. Facial key point detection: local explainability of A^* outputs

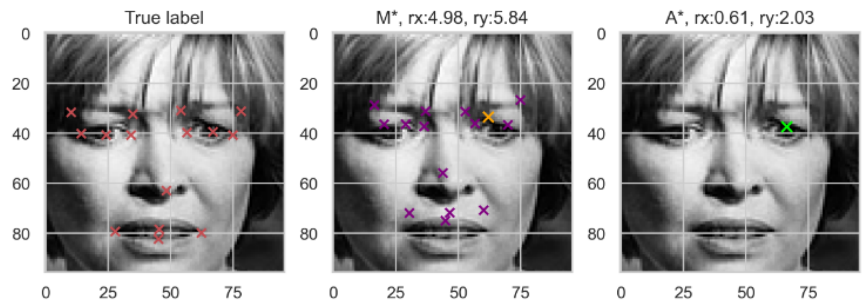

Figure 3: Test example for which we want to explain A^* outputs. True residuals are indicated for each coordinate related to the left eye center: rx and ry.

Table 3: True coordinate, M^* prediction and $A*$ (residual) prediction on a specific instance of the test dataset.

Target variable	True coordinate	M^* prediction	$A^*(\hat{\boldsymbol{y}})$ residual prediction
left eye center x	66.94	61.96	4.37
left eye center y	39.35	33.51	3.81

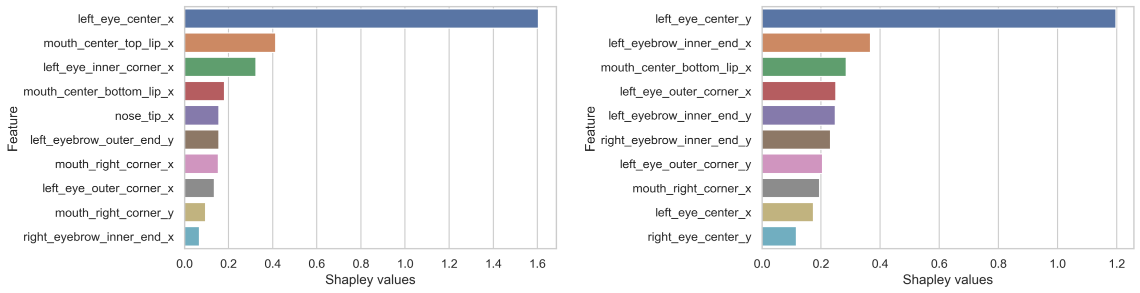

Figure 4: Top 10 Shapley values when AutoCorrect Model $A^*(\hat{\boldsymbol{y}})$ predicts the residuals for the left eye center x coordinate (left) and the left eye center y coordinate (right), respectively. The features (y-axis) correspond to M^* coordinate predictions leveraged by $A^*(\hat{\boldsymbol{y}})$.