Surrogate Models for Pipeline Recommendation in Multi-Label Classification

Leonardo D. Demore[1], **Aline Marques Del Valle**[2]
Rafael Gomes Mantovani[3], **Ricardo Cerri**[1]

[1] Institute of Mathematics and Computer Science (ICMC)
University of São Paulo (USP), São Carlos – SP – Brazil

[2]Federal Inst. of Education, Science and Technology of the South
of Minas Gerais (IFSULDEMINAS), Muzambinho – MG – Brazil

[3]Federal University of Technology – Paraná (UTFPR), Apucarana – PR –Brazil

leodemore@usp.br, aline.valle@ifsuldeminas.edu.br
rafaelmantovani@utfpr.edu.br, cerri@icmc.usp.br

Abstract. *Multi-Label Classification (MLC) is a challenging task where an instance is simultaneously classified into two or more classes. Due to its possible high dimensionality, feature selection is a crucial step in the classification pipeline. While AutoML can be used to optimize this pipeline selection, its empirical evaluation incurs prohibitive computational costs. This paper proposes a surrogate-based approach for efficient pipeline recommendation in MLC, comprising both classification and feature-selection algorithms. For this, a Multi-Output regressor model, trained on optimization histories, simultaneously predicts F1-score and model size of candidate pipelines. Coupled with an optimization algorithm, this method explores the search space without executing configurations in real time. Results prove the surrogate's high predictive accuracy (*$R^2 > 0.90$ *for performance and* > 0.98 *for structural cost) and its practical value. By applying this prior knowledge, the multi-objective approach identifies competitive configurations that greatly reduce feature selection time (e.g., from 280s to 0.15s) while recommending pipelines with a memory footprint orders of magnitude smaller (under 1 MB). Ultimately, this optimal trade-off makes AutoML viable in scenarios with severe temporal and storage restrictions.*

1. Introduction

Multi-Label Classification (MLC) allows a single instance to be associated with multiple labels simultaneously [Tsoumakas and Katakis 2007], frequently involving high-dimensional spaces. Consequently, feature selection has become a critical step in the classification pipeline, essential for removing redundant or noisy attributes, mitigating overfitting, and reducing computational costs [Theng and Bhoyar 2024]. However, the complex interdependence among labels, combined with the vast array of algorithms and their possible hyperparameter configurations, makes the manual design and tuning of these pipelines an exhaustive and often unfeasible task.

To overcome this complexity, Automated Machine Learning (AutoML) has been employed to optimize the entire process, from preprocessing to final classifier configuration. Despite its proven effectiveness, the application of AutoML faces a critical obstacle:

the extremely high computational cost of empirically training and evaluating thousands of pipeline combinations [He et al. 2021]. In MLC, an exhaustive search for the optimal pairing of feature selection algorithms and classifiers is computationally prohibitive. Although surrogate models are widely used in the optimization literature to estimate the performance of expensive functions [Bhosekar and Ierapetritou 2018], there remains a significant scientific gap regarding the use of surrogate-guided AutoML that also incorporates a feature-selection stage in multi-label scenarios.

To fill this gap, this paper proposes an AutoML approach that integrates surrogate models to evaluate and rank MLC pipelines. The solution employs a multi-output XGBoost regressor [Chen and Guestrin 2016], trained on a comprehensive meta-dataset of prior experimental evaluations. Specifically, the data originates from execution histories generated by the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [Del Valle et al. 2023] across multi-label benchmark datasets. The historical data comprises thousands of previously evaluated individuals, each representing a complete pipeline configuration (specific feature selection algorithms, base classifiers, and their hyperparameters) paired with its observed performance. The model simultaneously predicts two metrics of the proposed configuration: the F1-score and the final model size. The Optuna optimization framework [Akiba et al. 2019] was used in two distinct phases of the methodology: first, to fine-tune the XGBoost surrogate model's hyperparameters; and as the search algorithm to explore the combinatorial space and generate new multi-label classification pipelines. Thus, it becomes possible to estimate the performance of complex feature-selection and classification configurations without empirical evaluation.

2. Related Work

The traditional taxonomy of MLC distinguishes between *problem transformation* methods, which reduce the multi-label task into a collection of binary or multi-class subproblems (e.g., Binary Relevance, Classifier Chains, Label Powerset), from *algorithm adaptation* methods, which modify existing learners to natively handle label sets (e.g., ML-kNN, BPMLL) [Zhang and Zhou 2014]. Both families face a common structural challenge: the label space can be high-dimensional, and the labels are mutually dependent, meaning that the optimal MLC pipeline often exhibits complex hierarchical structures in which multi-label classifiers recursively nest other classifiers together with their preprocessing components. This complexity makes the MLC pipeline configuration space several orders of magnitude larger than its single-label counterpart, rendering manual selection and tuning of pipelines impractical even for domain experts [Wever et al. 2021].

In Single-label Classification (SLC), AutoML has matured into a well-established research area (Auto-WEKA, auto-sklearn, TPOT), with surrogate-based optimization emerging as the dominant strategy to mitigate the cost of empirical pipeline evaluation [Hutter et al. 2011, Feurer et al. 2015]. Methods such as Gaussian Processes and Random Forests act as cheap-to-query approximators of the true performance function, allowing Bayesian Optimization to focus evaluations on the most promising regions of the configuration space. Going beyond single-model surrogates, AVATAR [Nguyen et al. 2020] and AMLP [Palmes et al. 2023] apply this idea to full pipelines using Petri-net and signature-based surrogates, respectively, with AMLP matching the performance of methods with four-hour time budgets in under five minutes. Similarly, RankML [Laadan et al. 2019] pre-ranks pipelines via an XGBoost ranker trained

on meta-features derived from prior experiments, achieving accuracy comparable to exhaustive search at a fraction of the computational cost. However, all these approaches target single-output, single-label tasks and do not accommodate MLC's structural or multi-objective complexity.

In MLC, research on automated algorithm selection and pipeline configuration remains comparatively sparse. Early work [Wever et al. 2019] extended the hierarchical planning framework ML-Plan to the multi-label setting by incorporating MEKA's multi-label classifiers, laying the groundwork for systematic AutoML exploration of MLC search spaces. Subsequent work [de Sá et al. 2018] proposed the first grammar-based genetic programming approach to automatically select and configure MLC algorithms, treating the Combined Algorithm Selection and Hyperparameter Optimization (CASH) problem in MLC through evolutionary computation. [Moyano et al. 2020] built upon this evolutionary tradition by developing methods that jointly optimize MLC classifiers and feature selection strategies through multi-objective search. More recently, NSGA-II-based approaches [Del Valle et al. 2023] have explicitly framed MLC pipeline optimization as a multi-objective problem, simultaneously trading off predictive quality against model complexity. While these evolutionary methods successfully navigate multi-objective trade-offs and accommodate the hierarchical structure of MLC pipelines, they share a fundamental limitation: every candidate configuration generated during the evolutionary process must be instantiated and empirically evaluated on the target dataset. In scenarios involving dense multi-label data with high label cardinality, this empirical evaluation phase is a prohibitive computational bottleneck that severely limits the number of configurations that can be explored within practical time budgets.

The works reviewed above reveal a clear gap at the intersection of surrogate-assisted optimization and MLC. On one hand, surrogate models and meta-learning approaches have demonstrated their ability to dramatically reduce the evaluation cost of AutoML pipelines in the single-label setting [Nguyen et al. 2020, Palmes et al. 2023, Laadan et al. 2019]; on the other hand, the automated exploration of MLC pipelines has been exclusively pursued through computationally expensive empirical evaluation loops [Wever et al. 2021, Moyano et al. 2020, Del Valle et al. 2023]. To the best of our knowledge, no existing work has combined surrogate-based prediction with the multi-objective optimization of MLC pipelines. The present work fills this gap by training a Multi-Output XGBoost surrogate model on a corpus of historical MLC pipeline executions, enabling the simultaneous prediction of both the F1-score and the structural complexity of unseen pipeline configurations. By completely bypassing real-time empirical evaluations during the search phase, the proposed approach can explore a far larger portion of the MLC configuration space within practical time constraints, recommending competitive pipelines that conventional exhaustive evolutionary methods would fail to evaluate in the same time budget.

3. A Surrogate-Based Approach for Pipeline Recommendation in MLC

The methodology developed in this work aims to automate pipeline recommendations for MLC using surrogate models. Frequently formulated as a CASH problem, the challenge is to find the optimal combination of feature selection tools and classifiers that maximizes predictive performance while minimizing computational cost. To achieve this, the proposal formalizes an execution flow that replaces exhaustive empirical configuration eval-

uation with fast predictions. This substitution enables efficient exploration of the search space while handling the complex interdependence among multiple labels in a scalable manner. The complete flow of the proposed architecture is illustrated in Figure 1.

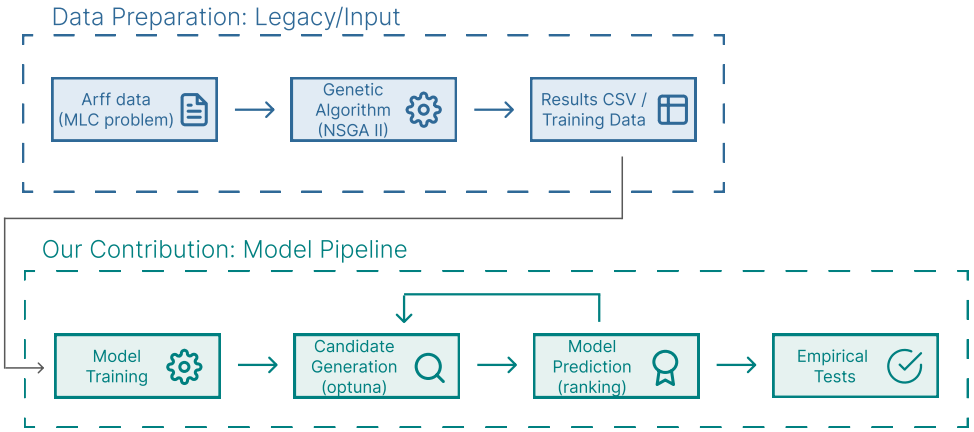


Figure 1. Complete flow of the proposed methodology, divided into the data preparation stage and the surrogate-based pipeline optimization cycle.

As illustrated in the Data Preparation sub-task, the surrogate model training require data from previous learning tasks (meta-data). The starting point consists of optimization records generated by the NSGA-II algorithm, as implemented in [Del Valle et al. 2023], in which each individual represents a pipeline comprising an optional feature-selection step and a mandatory base classifier. The structural components of each evaluated individual, alongside its performance targets (F1-score and model size), are systematically extracted and translated into a single tabular record. This history was built by executing the algorithm on five benchmark multi-label datasets from the MULAN[1] repository. Each execution generated a CSV dataset containing the evaluated hyperparameters and their corresponding performance metrics, serving as the supervised knowledge base for the metamodel.

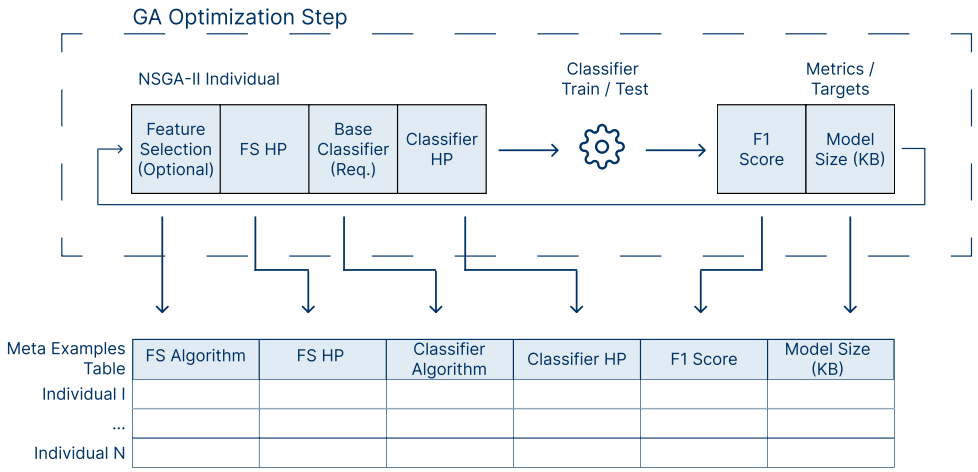


Figure 2. Mapping from NSGA-II execution to tabular meta-example format.

To enable supervised learning, the configurations in the histories are mapped to

[1]https://mulan.sourceforge.net/index.html

a structured matrix representation as illustrated by Table 1. A feature vector encodes all available classifiers and a comprehensive suite of feature selection algorithms, alongside their respective hyperparameter values. This feature selection pool encompasses native multi-label methods (IGMF, SCLS, LRFS, LSMFS, and MLSMFS), as well as classical techniques adapted through problem transformation strategies, specifically Binary Relevance (BR) and Pruned Problem Transformation (PPT). These adapted variants include Mutual Information (PPT_MI), SelectKBest (BR_SelectKBest, PPT_SelectKBest), ReliefF (BR_ReliefF, PPT_ReliefF), Recursive Feature Elimination (Sklearn RFE, PPT_RFE), and SelectFromModel (Sklearn SelectFromModel, YY_SelectFromModel)[2]. A conditional encoding strategy is employed: when an algorithm is selected for the pipeline, its hyperparameters take their actual numerical values; otherwise, they are set to null values (*NaN*), indicating that the corresponding branch is inactive in the hyperspace. The vector is complemented by categorical control variables that indicate preprocessing decisions, such as whether to apply data normalization and whether to bypass feature selection. As target variables (outputs), the surrogate model is trained to predict two pipeline metrics: the *F1-score* and the generated log model size. Specifically, model size refers to the disk size (in kilobytes) of the trained classifier after serialization and, due to its wide dynamic range across configurations, this variable is log-transformed (using $\log(1 + x)$) prior to training the surrogate model.

Table 1. Representation of the sparse meta-dataset encoding MLC pipelines.

Pipeline ID	Control Variables		Feature Selection (Sparse)			Classifier (Sparse)			Targets	
	Norm.	Bypass FS	FS Alg.	`IGMF_k`	`PPT_MI_th`	Clf Alg.	`RF_est`	`SVM_C`	F1-score	Log Size
1	0	*NaN*	IGMF	50	*NaN*	RF	100	*NaN*	0.682	14.2
2	*NaN*	0	*None*	*NaN*	*NaN*	SVM	*NaN*	1.5	0.514	15.8
3	0	*NaN*	PPT_MI	*NaN*	0.05	RF	50	*NaN*	0.640	13.5
...	...	...	...	...	...	...	...	...	...	...

The core of the proposed architecture consists of the XGBoost estimator wrapped in a multi-output strategy. The definition of this configuration was based on rigorous preliminary experiments. Initially, the performance of XGBoost was compared with other popular estimators (RF, LightGBM, and Multilayer Perceptron - MLP), distinguishing itself for its native ability to handle high data sparsity and efficiently handle null values without the need for complex imputation techniques. To handle the problem's multi-objective nature, three MLC approaches were evaluated: Regressor Chain, Native Adaptation, and Independent Prediction (Multi-Output). For the chain and independent strategies, we used the `RegressorChain` and `MultiOutputRegressor` meta-estimator wrappers provided by the *scikit-learn* library.

The chain wrapper explicitly models dependencies between targets by linking predictions sequentially, where the output of one regressor serves as an input feature for the next. Conversely, the independent wrapper treats each target as a separate, isolated regression task, estimating them concurrently without assuming any mutual correlation. In contrast to these wrapper-based methods, Native Adaptation refers to the inherent capacity of specific algorithms to naturally predict multiple targets simultaneously within a single unified model structure, without requiring external encapsulation. Although the

[2]A detailed description of the feature selection methods is provided in [Del Valle 2025].

chain strategy leverages target correlations to guide the regression, the independent architecture was ultimately chosen. Preliminary results justify this technical decision: despite the existence of a moderate correlation between the *F1-score* and the model size (with Spearman's rank correlation ρ varying between 0.29 and 0.55, as illustrated by the scatter plots in Figure 3), the chain and independent strategies achieved practically identical predictive scores during validation. Given this performance parity, the independent approach was selected because it guarantees a substantial reduction in training time and optimizes the computational cost of building the surrogate model.

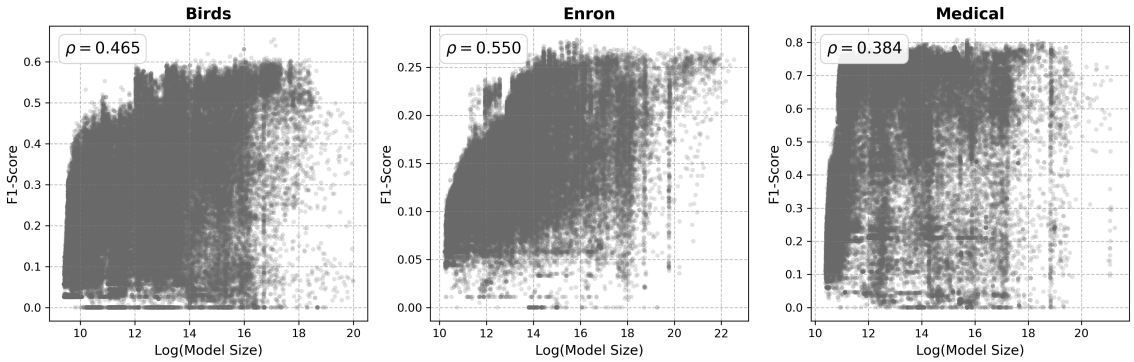


Figure 3. Scatter plots illustrating the moderate monotonic correlation (Spearman's ρ) between Log (Model Size) and *F1-Score* across the datasets.

Once the surrogate model is induced, it is integrated with the Optuna framework to efficiently search for and rank new unseen pipelines. The mechanics of this integration operate as a feedback loop: during the optimization process, Optuna employs the Tree-structured Parzen Estimator (TPE) sampler to navigate the hyperspace and propose a candidate configuration, which comprises specific algorithms and their hyperparameters. Instead of instantiating and empirically evaluating this proposed pipeline, Optuna feeds it directly into the trained surrogate. The surrogate model then instantly "scores" the candidate by predicting its objective metrics (F1-score and model size). Finally, Optuna uses these predicted scores to update its probabilistic model, guiding the TPE sampler toward increasingly promising regions of the hyperspace without ever executing the actual ML routines. The search can be configured in single-objective mode, aiming exclusively for the highest predicted F1-score, or in multi-objective mode, using the Pareto frontier to balance predictive performance and model size. At the end of the iterations, all the pipelines evaluated by the surrogate are ranked, and only the top-K candidates are selected for real execution. This strategy saves processing power and directs computational resources strictly to the most promising regions of the search space.

4. Methodology

4.1. Dataset and Preprocessing

We evaluated five widely used multi-label datasets from the MULAN repository: *birds*, *medical*, *enron*, *scene*, and *yeast*. They cover a variety of domains, including natural language processing, computer vision, and biology, and present varying levels of cardinality, density, and dimensionality. All these datasets underwent thorough cleaning and

preprocessing. Initially, duplicated examples and unlabeled instances were removed; minority labels (with ten or fewer instances) and constant features were eliminated to prevent learning inconsistencies. After, the datasets were partitioned using a 3-fold stratified multi-label cross-validation (CV) [Sechidis et al. 2011]. It is important to emphasize that this partitioning was employed before the generation of the meta-dataset: each candidate pipeline in the legacy database was evaluated across these three folds to obtain a robust, unbiased estimate of its predictive performance (e.g., F1-score) before being recorded as a meta-example. It is important to mention that all F1-score values correspond to the macro-averaged F1-score, computed by averaging the per-label F1-scores with equal weight regardless of label frequency, which is particularly relevant given the label imbalance observed across the evaluated datasets (table 2). All preprocessing tasks were conducted using the Pandas[3] and Scikit-Multilearn[4] libraries.

Finally, the preprocessed datasets were exported in ARFF format. Regarding the feature space, the exported datasets were handled differently based on their structural sparsity. For dense datasets, where the majority of feature values are non-zero, two structural versions were maintained: one with the data in its original scale and another with normalized features. This dual representation was provided because many ML algorithms, such as distance-based classifiers are highly sensitive to feature scaling. Conversely, for sparse datasets (typically containing a vast majority of zero values), normalization was strictly avoided. Applying standard normalization to sparse matrices is highly problematic, as it destroys the sparsity structure by converting zero values into non-zeros, which dramatically increases memory consumption and compromises data interpretability. Table 2 summarizes the statistical characteristics of the datasets after preprocessing, detailing the source domain, the number of instances, features, and labels, as well as label cardinality and density.

Table 2. Multi-label datasets used in the experiments after pre-processing.

Dataset	Domain	Instances	Features	Labels	Cardinality	Density
birds	Audio/Biology	337	260	15	1.84	0.12
medical	Clinical Text	942	1449	20	1.20	0.06
enron	Text	1689	1001	42	3.35	0.08
scene	Image	2402	294	6	1.07	0.17
yeast	Biology	2417	103	14	4.24	0.30

4.2. Experimental Setup

To ensure reproducibility of the results, the experiments were structured into two phases: i) surrogate model training; and ii) the pipeline recommendation phase. In the first, the surrogate model was trained independently for each of the five datasets. The data was split into training (80%) and testing (20%) sets and, to achieve the best fit regarding both target metrics, the XGBoost regressor's hyperparameters[5] were fine-tuned using the Optuna framework. For each candidate configuration, a 3-fold CV was performed on the training set, and the objective function to be minimized was a weighted combination of

[3] `https://pandas.pydata.org/`

[4] `https://github.com/scikit-multilearn/scikit-multilearn`

[5] Including *n_estimators*, *learning_rate*, *max_depth*, and regularization terms, among others.

the Root Mean Squared Error (RMSE) obtained for both targets. We assigned greater importance to the F1-score prediction (weight of 0.9) relative to the log-transformed model size prediction (weight of 0.1). During the prior stage that generated the training data, a computational wall-time of 300 seconds was established for executing any feature selection algorithm and 600 seconds for executing any classifier; processes that exceeded these limits were aborted and penalized. Furthermore, for datasets that included this option (such as birds, scene, and yeast), both the original and normalized versions of the features were evaluated.

In the recommendation phase, the surrogate model acts as the evaluator. The exploration of the combined hyperspace, comprising 60 algorithms (feature selection and classification) and a total of 145 hyperparameters, was conducted by Optuna and its TPE estimator in multi-objective mode, simultaneously maximizing the predicted F1-score and minimizing the predicted model size. To ensure a comprehensive search, the optimizer was configured to run $100,000$ trials, a process that required approximately 2 to 4 hours of wall-clock time given the near-instantaneous inference cost of the surrogate model. In this flow, each proposed configuration was submitted to the XGBoost surrogate for inference, which instantly predicted its performance and cost, enabling the construction of the predicted Pareto frontier. From this Pareto-optimal set, the top-10 pipelines with the highest F1-score were selected as the final recommendations for each dataset. Crucially, to validate the optimization process, these Pareto-optimal pipelines were subsequently instantiated, fully trained, and evaluated on the target datasets. This final empirical execution allowed for a direct comparison between the surrogate's predicted metrics and the actual observed performance, thereby confirming the reliability and practical viability of the recommended configurations.

5. Experiments and Discussion

5.1. Surrogate Model Performance

Before employing the surrogate model as an evaluator in the recommendation process for new pipelines, it is imperative to validate its ability to learn and map the correlation between algorithm configurations and target metrics. Due to the wide range in magnitude, the classifier size was modeled and evaluated on a logarithmic scale. The multi-output XGBoost regressor was evaluated on the historical training data using the Coefficient of Determination (R^2) and the RMSE for both objectives: the *F1-score* (original scale) and the model size. Table 3 summarizes both metrics of the single, fully-optimized surrogate model evaluated on the hold-out data for all datasets.

Table 3. Predictive performance of the surrogate model. RMSE and R^2 are reported for both F1-score and the logarithm of Model Size.

Dataset	F1-Score		Model Size (log)	
	RMSE	R^2	**RMSE**	R^2
birds	0.0397	0.9281	0.1100	0.9972
enron	0.0155	0.9159	0.2700	0.9856
medical	0.0411	0.9642	0.1700	0.9934
scene	0.0344	0.9423	0.1700	0.9946
yeast	0.0218	0.9083	0.2200	0.9911

The results demonstrate a strong predictive capacity across all scenarios. For the

F1-score prediction, the model obtained an R^2 consistently above 0.90, reaching a peak of 0.9642 on the medical dataset, accompanied by remarkably low RMSE values. This indicates that XGBoost successfully captured the complex nonlinear interactions between the feature selection hyperparameters and the classifiers. Additionally, the model size prediction exhibited near-deterministic behavior, with R^2 exceeding 0.98 across all datasets. This statistical robustness validates the regressor as a reliable estimator to guide the hyperparameter search in the next phase, ensuring that the predictions used to construct the Pareto frontier accurately reflect the expected real-world performance.

5.2. Pipeline Recommendation Evaluation

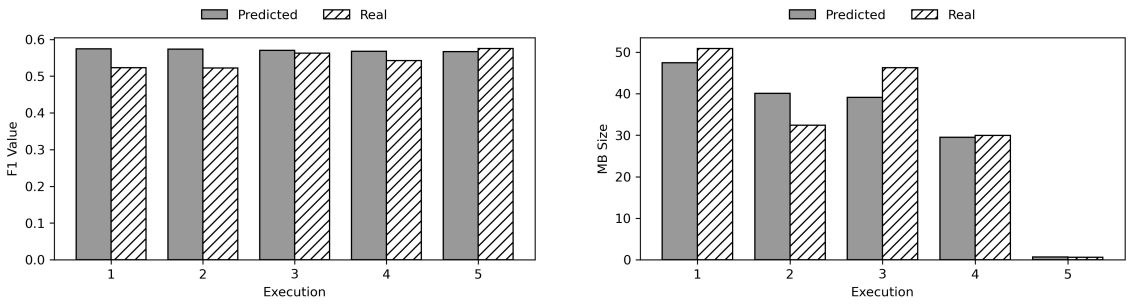
After the search phase conducted by the Optuna optimizer, the top-10 pipelines with the highest *F1-score* for each dataset were selected for actual execution. The objective of this step is to compare surrogate model predictions with actual results to validate the recommendation's efficacy. Each configuration was instantiated and evaluated using a 3-fold stratified CV, recording the average F1-score, the classifier size (in megabytes), and the execution times, subdivided between the feature selection phase and the base model training. Table 4 summarizes these results, comparing predicted with the real performance values for each dataset.

Table 4. Comparison between surrogate predictions and real results (averaged across the top-10 recommended pipelines and 3 cross-validation folds).

Dataset	F1-Score		Model Size (MB)	
	Predicted	**Real (Mean $\pm$ SD)**	**Predicted**	**Real (Mean $\pm$ SD)**
birds	0.560	0.533 ± 0.028	15.85	16.29 ± 21.25
enron	0.252	0.245 ± 0.010	9.21	7.46 ± 10.63
medical	0.774	0.721 ± 0.028	9.69	11.25 ± 17.45
scene	0.783	0.772 ± 0.016	13.29	14.12 ± 23.50
yeast	0.481	0.476 ± 0.018	2.68	2.51 ± 2.74

The obtained results confirm the utility of the surrogate model for pipeline recommendation and reveals predictive behaviors that warrant attention. We observe a systematic tendency for the regressor to slightly overestimate predictive performance and marginally underestimate the final model size relative to empirical behavior. On the birds dataset, for instance, while the model predicted an average *F1-score* of 0.560, the actual mean obtained across the folds was 0.533. This deviation, inherent to approximating the objective function in the search space, does not invalidate the ranking: the selected pipelines consistently remained among the most competitive in their respective datasets.

The comparison between surrogate model predictions and real tests demonstrates the effectiveness of the multi-objective optimization approach. Figures 4(a) and 4(b) show the *F1-score* and model size for the top-5 ranked pipelines on the birds dataset. The behavior observed in Execution 5 is particularly noteworthy. While the surrogate model accurately predicted performance (achieving the highest real *F1-score* among the candidates), it also recommended a pipeline with a memory footprint that is orders of magnitude smaller than the others (less than 1 MB). This result highlights the capability of the Multi-Output XGBoost in guiding the optimizer toward regions of the search space that offer the optimal trade-off between high predictive capacity and low storage cost.



(a) F1-Score Predicted vs. Real **(b) Model Size Predicted vs. Real**

Figure 4. Comparative analysis of the top-5 recommended pipelines for the *birds* dataset, illustrating the trade-off efficiency.

However, the most significant finding lies in the substantial reductions in computational resource consumption and execution time. Scanning the recommendations, we identified pipelines that achieve nearly real *F1-scores* but differ substantially in their feature-selection approaches. By ranking the solutions using the surrogate model's response surface, the need to execute computationally expensive configurations was avoided. To illustrate this efficiency gain, Table 5 presents a case study extracted from the recommendations for the birds dataset. Two distinct pipelines ranked at the top of the surrogate model's Pareto frontier were selected.

Table 5. Case study on the *birds* dataset illustrating time efficiency. Different Feature Selection (FS) approaches yield similar predictive performance but drastically different execution times.

Pipeline	FS Algorithm	F1-Score (Real)	FS Time (s)	Classifier Time (s)
Pipeline A	*PyIT_LSMFS*	0.523	277.40	8.56
Pipeline B	*BR_SelectKBest*	0.522	0.15	8.72

As demonstrated in Table 5, although both pipelines use classifiers with configurations that deliver a competitive average F1-score close to 0.52, and consume virtually the same time in the classification stage (around 8.5 seconds), the time invested in preprocessing differs significantly. Pipeline A consumed an average of 277.4 seconds to perform feature selection, bordering the 300-second timeout limit stipulated in the experiments. In contrast, Pipeline B achieved equivalent efficacy by executing the preprocessing in just 0.15 seconds. In a classic AutoML scenario, the exhaustive evaluation of configurations similar to Pipeline A would consume many processing hours. The use of the surrogate model bypassed this bottleneck, isolating and recommending highly time-efficient routes without sacrificing predictive power.

Despite significant gains in temporal and storage efficiency, the analysis revealed a limitation in the current exploration strategy. By comparing the surrogate model's highest F1-score to the performance ceiling in the training data, we observed that the method tends to converge to solutions slightly inferior to the global optimum. This suggests that the Optuna optimizer may be prematurely converging to local optima. Guided solely by the XGBoost's point predictions, the search process tends to prioritize exploiting response surface regions already identified as promising, neglecting the exploration of isolated configurations that could harbor pipelines with superior performance.

6. Conclusions and Future Works

In this work we proposed and validated surrogate models to efficiently recommend algorithms and hyperparameter configurations. By using a Multi-Output XGBoost regressor coupled with the Optuna optimizer, it was possible to map the hyperspace and simultaneously predict both predictive performance (F1-score) and structural cost (model size) without instantiating each processing routine.

Experiments demonstrated the high reliability of the proposed model. The surrogate estimator achieved R^2 values consistently above 0.90 for the F1-score prediction and over 0.98 for the logarithmic estimation of the classifier size. By applying this prior knowledge in the ranking phase, the multi-objective approach demonstrated its practical value: it successfully identified underrepresented configurations that achieved competitive empirical performance (around 0.52 on the birds dataset) while drastically reducing the time spent on feature selection from nearly 280 seconds to 0.15 seconds. This optimization saves valuable processing resources and makes AutoML viable in scenarios with critical temporal constraints.

For future work, we will focus on strengthening the empirical validation of the approach, enhancing its generalization capability and search space exploration, including different datasets in the benchmark. In the longer term, we aim to evolve the system into a Zero-Shot AutoML scenario, enabling pipeline recommendations for entirely unseen datasets without retraining it. We also envision including execution time as a third predictive target, extending the methodology to Extreme Multi-Label Classification (XMLC) domains, and conducting interpretability studies to elucidate how hyperparameters affect the generated predictions.

Acknowledgments

The authors would like to thank the grants #2024/19234-6 and #2025/15370-5 from São Paulo Research Foundation (FAPESP), and the National Council for Scientific and Technological Development – Brazil (CNPq), grant 300934/2025-4.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 2623–2631.

Bhosekar, A. and Ierapetritou, M. (2018). Advances in surrogate based modeling, feasibility analysis, and optimization: A review. *Computers & Chemical Engineering*, 108:250–267.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. ACM.

de Sá, A. G. C., Freitas, A. A., and Pappa, G. L. (2018). Automated selection and configuration of multi-label classification algorithms with grammar-based genetic programming. In *Parallel Problem Solving from Nature*, volume 11101, pages 308–320.

Del Valle, A. M. (2025). *Aprendizado de máquina automatizado multiobjetivo para classificação multirrótulo*. Tese (doutorado em ciência da computação), Universidade Federal de São Carlos, São Carlos.

Del Valle, A. M., Mantovani, R. G., and Cerri, R. (2023). AutoMMLC: An Automated and Multi-objective Method for Multi-label Classification. In *Intelligent Systems*, pages 291–306.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. (2015). Efficient and Robust Automated Machine Learning. In *Advances in Neural Information Processing Systems*, volume 28.

He, X., Zhao, K., and Chu, X. (2021). AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212:106622.

Hutter, F., Hoos, H. H., and Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization*, volume 6683, pages 507–523.

Laadan, D., Vainshtein, R., Curiel, Y., Katz, G., and Rokach, L. (2019). RankML: a Meta Learning-Based Approach for Pre-Ranking Machine Learning Pipelines.

Moyano, J. M., Gibaja, E. L., Cios, K. J., and Ventura, S. (2020). Combining multi-label classifiers based on projections of the output space using evolutionary algorithms. *Knowledge-Based Systems*, 196:105770.

Nguyen, T.-D., Maszczyk, T., Musial, K., Zöller, M.-A., and Gabrys, B. (2020). Avatar - machine learning pipeline evaluation using surrogate model. In *International Symposium on Intelligent Data Analysis*, page 352–365.

Palmes, P. P., Kishimoto, A., Marinescu, R., Ram, P., and Daly, E. (2023). Designing machine learning pipeline toolkit for automl surrogate modeling optimization. *Proceedings of the JuliaCon Conferences*, 1(1):129.

Sechidis, K., Tsoumakas, G., and Vlahavas, I. (2011). On the Stratification of Multi-label Data. In *Machine Learning and Knowledge Discovery in Databases*, pages 145–158.

Theng, D. and Bhoyar, K. K. (2024). Feature selection techniques for machine learning: a survey of more than two decades of research. *Knowledge & Information Systems*, 66(3):1575–1637.

Tsoumakas, G. and Katakis, I. (2007). Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3):1–13.

Wever, M., Mohr, F., and Hüllermeier, E. (2019). Automating multi-label classification extending ml-plan. In *ICML 2019 Workshop on Automated Machine Learning*.

Wever, M., Tornede, A., Mohr, F., and Hüllermeier, E. (2021). AutoML for multi-label classification: Overview and empirical evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9):3037–3054.

Zhang, M.-L. and Zhou, Z.-H. (2014). A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837.