AFRL-RI-RS-TR-2024-053

ACCESSIBLE MACHINE LEARNING

CARNEGIE MELLON UNIVERSITY

MAY 2024

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2024-053 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
FRANCO MILVIO
Work Unit Manager

/ S /
JULIE BRICHACEK
Chief, Information Systems Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
		START DATE	END DATE
MAY 2024	FINAL TECHNICAL REPORT	MARCH 2017	DECEMBER 2023

4. TITLE AND SUBTITLE

ACCESSIBLE MACHINE LEARNING

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
FA8750-17-2-0130	N/A	62702E

5d. PROJECT NUMBER	5e. TASK NUMBER	5f. WORK UNIT NUMBER
		R27R

6. AUTHOR(S)

Artur Dubrawski

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Carnegie Mellon University 5000 Forbes Ave. Pittsburgh PA 15213-3815	

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Air Force Research Laboratory/RISC 525 Brooks Road Rome NY 13441-4505	AFRL/RI	AFRL-RI-RS-TR-2024-053

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The project aims to develop an automated, interactive model discovery system that will enable users with no data science background to create and maintain useful models of complex processes, and that will increase effectiveness of competent data scientists via automation of multiple component tasks.

15. SUBJECT TERMS

An automated, interactive model discovery framework, Machine learning on complex objects, Automated composition of complex models, Dialog-capable Data Science Agent, Hyper-vised and Hyperactive Learning, Explainable Machine Learning and human interpretability of data, models, and results

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	C. THIS PAGE		
U	U	U	SAR	21

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
MILVIO FRANCO	**N/A**

Table of Contents

List of Figures

1.0 SUMMARY

Throughout the course of this project, our team completed its goals by investigating and developing new machine learning algorithms, interaction protocols and abstract representations, identifying the optimal level of abstraction of data, models and information exchanged between data, models and the users, and by integrating our research findings in the development of our open-source Auto^{n}ML software under D3M framework. We have broadly disseminated our results through publications and presentations.

Under Technical Area 1 (TA1), our team contributed to the software development of the D3M framework and machine learning primitives as well as the evaluation framework. We proposed novel scalable Bayesian Optimization algorithms to improve higher dimensional multi-fidelity evaluations, and developed Dragonfly, an open source Python library for scalable and robust Bayesian Optimisation. For time series data, we proposed and built novel time series forecasting algorithms NBEATSx and NHiTS, which are currently State-Of-The-Art of deep neural forecasting, and we developed NeuralForecast, an open software package with a number of deep-learning based forecasting models for time series data.

Under TA2, our team developed our own open-source automated machine learning tool, Auto^{n}ML, under the D3M framework. Our Auto^{n}ML consistently ranked top in DARPA evaluation leaderboards across different development and evaluation phases. Auto^{n}ML supports regression, classification, forecasting, graph-based analytics of tabular, text, image, audio, and video data. It could run locally through Command Line Interface (CLI), Python IDEs, Docker, Amazon AWS platform and HPC environments with Jupyter Notebook interface. We demonstrated key capabilities of Auto^{n}ML in various real-world data analytics applications ranging from clinical research to predictive maintenance. We also conducted comprehensive performance evaluations of our Auto^{n}ML in comparison to noteworthy open-source AutoML tools such as H2O AutoML, Tree-Based Pipeline Optimization Tool (TPOT), AutoGluon and Auto-Sklearn using public OpenML datasets. Our Auto^{n}ML outperforms others in high-dimensional datasets and appears more consistent than alternatives across varying time budgets.

To adapt to the increasing demand of using large language models and pre-trained large models and resolve the dependency conflicts with new algorithms, we re-architectured Auto^{n}ML to remove the D3M framework dependencies and to improve the overall extensibility, scalability, and usability of the tool. The new Auto^{n}ML, ngAuto^{n}ML, is highly modularized with flexible customization for pipeline template, model catalog, metric catalog, and evaluation process. The architecture allows quick integration of new algorithms and metrics by both developers and end users. The "plugin" mechanism also allows structuring an deployment of different configurations of ngAuto^{n}ML with potentially conflicting software dependencies. ngAuto^{n}ML was released in both Gitlab source code and PyPI version for easy pip installation. We expect ngAuto^{n}ML to be of broad use as an efficiency booster tool for data scientists. We designed it with an ease of deployment and capacity for growth and inclusion of third party and user-provided components as the key principles. We are using it as a default first-use tool in our research team, and we will promote its broad use elsewhere. Essential parts of the re-engineered architecture of the system are also being used as building blocks for other machine learning and artificial intelligence tools being developed at the CMU Auton Lab.

2.0 INTRODUCTION

The objective of this project was to develop an automated, interactive model discovery framework that will enable users with no data science background to create and maintain useful models of complex processes, and that will increase effectiveness of competent data scientists via automation of multiple component tasks. This effort focused on five tightly-coupled research thrusts contributing to all DARPA D3M Technical Areas: (1) Machine learning on complex objects (TA1, it involves learning from and about objects at various levels of abstraction); (2) Automated composition of complex models (TA2, this thrusts focuses on identifying task-optimal compositions of complex models); (3) Dialog-capable Data Science Agent (TA3, enabling understanding of user requests and feedback and translating them into task orders for an machine learning (ML) system); (4) Hypervised and Hyperactive Learning (TA3, identifying the most appropriate levels of abstraction of data, models, and information exchanged between data, models and users); and (5) Explainable Machine Learning (TA3, this thrust involves enhancing human interpretability of data, models, and results).

The project has yield impact across several tech transfer paths including: 1. Academic publications, 2. Releases to the public domain of modular software integrable with other components of the D3M environment, 3. Collaboration with other D3M performers and DARPA partners, 4. Applications outside of D3M areas of immediate interest.

We have produced 39 publications including 6 journal papers, 17 prime computer science conference papers (including 6 at NeurIPS, and 3 each at ICML and AISTATS), and 3 Ph.D. theses.

We have produced multiple open source software libraries with large user bases. One of them (NeuralForecast) has been integrated in a line of commercial products by a successful startup company (Nixtla, https://www.nixtla.io/).

3.0 METHODS, ASSUMPTIONS, AND PROCEDURES

Current Machine Learning technology can be powerful, but its application often requires high level expertise of its users to be effective and successful. We intend to modify and extend the existing technology to remedy this challenge and make ML significantly more accessible to a wide range of users and therefore much more impactful.

Our main goals are to lower the thresholds of accessibility of powerful Machine Learning (ML) technology to non-expert users, and to remedy difficulties in its practical application that often impede productivity of data science experts in using ML. We achieved these goals by investigating new algorithms, interaction protocols, and abstract representations, while focusing on the following specific objectives and tasks: (1) Interpreting user's requests and automatically translating them into task orders for a ML system to execute; (2) Identifying the most appropriate levels of abstraction of data, models, and information exchanged between data, models and the users, to optimize utility of interactions, minimize required human labor, and maximize automation of tasks, while producing high quality results; (3) Devising new algorithms to learn from and about objects at various levels of abstraction; (4) Identifying task-optimal compositions of complex models; and (5) Enhancing human interpretability of data and models while helping to diagnose and remediate common ML

process faux pas. The current state-of-the art takes on the impediments to accessibility and productivity of ML selectively and sparingly. We will address these issues systematically and comprehensively. The expected end result is a significant expansion of the usability and pragmatic utility of ML technology and its wider and deeper penetration of problem spaces in a wide range of domains of human activity.

4.0 RESULTS AND DISCUSSION

4.1 TA1 - Machine Learning Capability Development

4.1.1. Development of ML Primitives and D3M Framework

We developed foundational algorithmic capabilities of both the individual primitives as well as pipeline building and evaluation engine with close communication and collaboration with other performers of the D3M project during its first phase. We participated in the design and development of the D3M framework and demonstrated the usability of this framework by presenting brief evaluation results of eight AutoML systems that use it [1]. We also contributed to the development of D3M developer documentation and D3M landing page development.

4.1.2. Dragonfly - Scalable Bayesian Optimization

Bayesian optimization can be used for automating optimization of black-box functions such as machine learning models whose evaluations are usually expensive. Our team developed Dragonfly, an open source Python library for scalable and robust Bayesian Optimization. Dragonfly incorporates multiple recently developed methods that allow Bayesian optimization to be applied in challenging real world settings; these include better methods for handling higher dimensional domains, methods for handling multi-fidelity evaluations when cheap approximations of an expensive function are available next to more precise but more expensive evaluations, methods for optimizing over structured combinatorial spaces, such as the space of deep neural network architectures, and methods for handling parallel evaluations [2, 3]. Beyond vanilla techniques, Dragonfly provides an array of tools to scale up Bayesian optimization to expensive large scale problems. These include features and functionality that are especially suited for high dimensional optimization tasks (optimizing for a large number of variables), parallel evaluations in synchronous or asynchronous settings (conducting multiple evaluations in parallel), multi-fidelity optimization (using cheap approximations to speed up the process), and multi-objective optimization (optimizing multiple objective functions simultaneously). The current software package and its documentation can be found at https://github.com/dragonfly/dragonfly.

4.1.3. NeuralForecast

NeuralForecast is a growing open collection of deep-learning based forecasting models that yield interpretable results and focus on robustness. The collection includes models ranging from simple neural networks such as multilayer perceptron (MLP) or recurrent neural networks (RNNs) configured to perform forecasting tasks, to novel Special Operations Team-Alpha (SOTA) contributions such as Neural Basis Expansion Analysis with exogenous variable (NBEATS), Neural Hierarchical Interpolation for Time Series Forecasting (NHiTS), Temporal Fusion Transformer (TFT), Exponential Smoothing-Recurrent Neural Networks (ES-RNN) and their extensions, developed by our research team. Implementation of the NeuralForecasting package allows for GPU and CPU code parallelization. The models are capable of point and probabilistic predictions. The

models can also incorporate various types of exogenous variables: historic, static and available at the time of making predictions. As an example, NBEATSx improves accuracy by 20% over NBEATS/ESRNN (SOTA generic alternatives) nn the Electricity Price Forecasting challenge, and by 5% over methods specialized to the task, while being interpretable [4]. NHiTS improves accuracy by 25% on the AAAI best paper with a transformer-based method with 50x less computation [5]. NHiTS is among the most popular SOTA deep forecasting techniques worldwide with hundreds of software downloads and several application-oriented publications using it as a tool of choice. The current software package and its documentation can be found at https://nixtlaverse.nixtla.io/neuralforecast/index.html It has 2.3 thousand stars and 268 forks on GitHub.

4.1.4. Coarse-to-Fine Curriculum Learning

When faced with learning challenging new tasks, humans often follow sequences of steps that allow them to incrementally build up the necessary skills for performing these new tasks. However, in machine learning, models are most often trained to solve the target tasks directly. Inspired by human learning, our team proposed a novel curriculum learning approach which decomposes challenging tasks into sequences of easier intermediate goals that are used to pre-train a model before tackling the target task [6]. We focused on classification tasks, and designed the intermediate tasks using an automatically constructed label hierarchy. We trained the model at each level of the hierarchy, from coarse labels to fine labels, transferring acquired knowledge across these levels. The result is a curriculum learning algorithm for classification that: (i) breaks down complex classification tasks into sequences of simpler tasks, and (ii) goes through these tasks in order of increasing difficulty, training classifiers and transferring acquired knowledge between the learned classifiers. We showed that our approach achieves significant performance gains on both synthetic and real data, using multiple neural network architectures. Finally, our approach is purely a training strategy, and does not incur any additional memory or computational costs during inference.

4.2. TA2 - Auto^{n}ML

4.2.1 Software Development

In the scope of TA2 efforts, we developed our own open-source automated machine learning tool, Auto^{n}ML, based on the D3M framework and software components developed under TA1. Auto^{n}ML multiplies capacity of Data Scientists by automating searches for plausible modeling process designs and supports regression, classification, forecasting, graph-based analytics of tabular, text, image, audio, and video data. Our Auto^{n}ML has consistently ranked top 1 in DARPA evaluation leaderboards (MARVIN leaderboard: https://marvin.datadrivendiscovery.org/leaderboard) across subsequent development and evaluation phases.

We identified various use cases and user requirements of an ideal Auto^{n}ML tool with our rich experience in prototyping real-world data analytics problems and advancing machine learning algorithms to their practical use. The functionality of Auto^{n}ML includes but not limit to the following aspects:

- Search, train and test pipelines for D3M datasets as well non-D3M datasets;
- Easily specify data tasks and convert raw datasets into the D3M format;
- Rank pipelines based on specified metrics; output predictions of both training and testing

data; export pipelines into executable python scripts with potential needs in adding customized non-D3M functions and tuning hyperparameters of D3M primitives;
- Define customized evaluation metrics;
- Define customized evaluation frameworks including cross-validation strategy and customized data fold splittings.

The pipeline search process in Auto^{n}ML is illustrated in Figure 1. A pipeline is basically a series of steps that are executed in order to solve a particular problem (such as prediction based on historical data). A step of a pipeline is usually a primitive that individually could, for example, transform data into another format, or fit a model for prediction. With a user-specified data task, including task type, evaluation metric and path to training and testing datasets, Auto^{n}ML will search for all combinations of primitives, train pipelines on training data with hyperparameter tuning and rank pipelines based on evaluation metric. The output of Auto^{n}ML includes a list of top pipelines, training and testing predictions, executable pipelines in .json files and python scripts. Pipelines could be ranked and compared visually as shown in Figure 2 and Figure 3 with PipelineProfiler package developed by D3M collaborator AlphaD3M team from New York University.

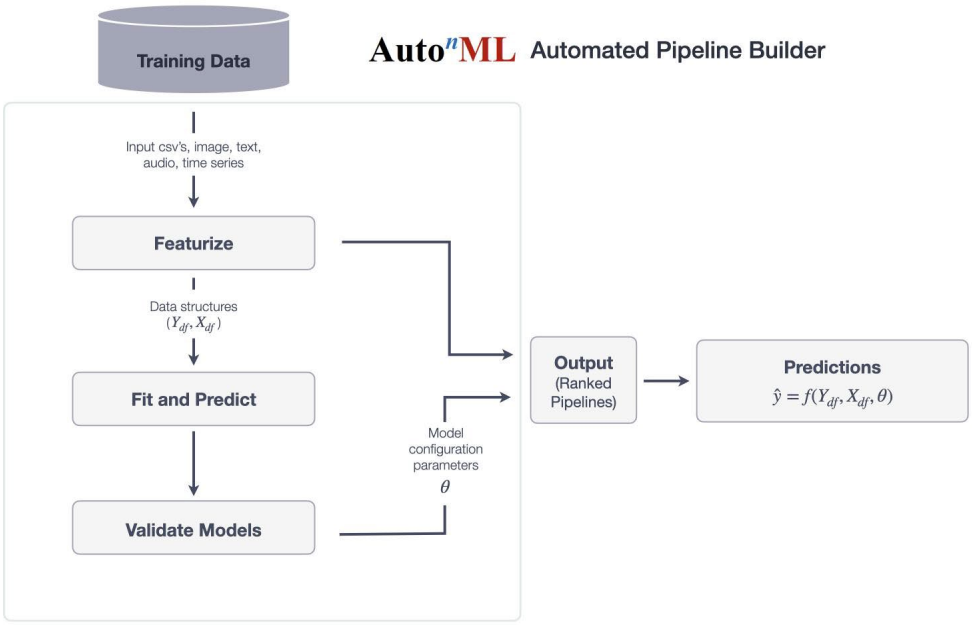

Figure 1: Auto^{n}ML Pipeline Search Process.

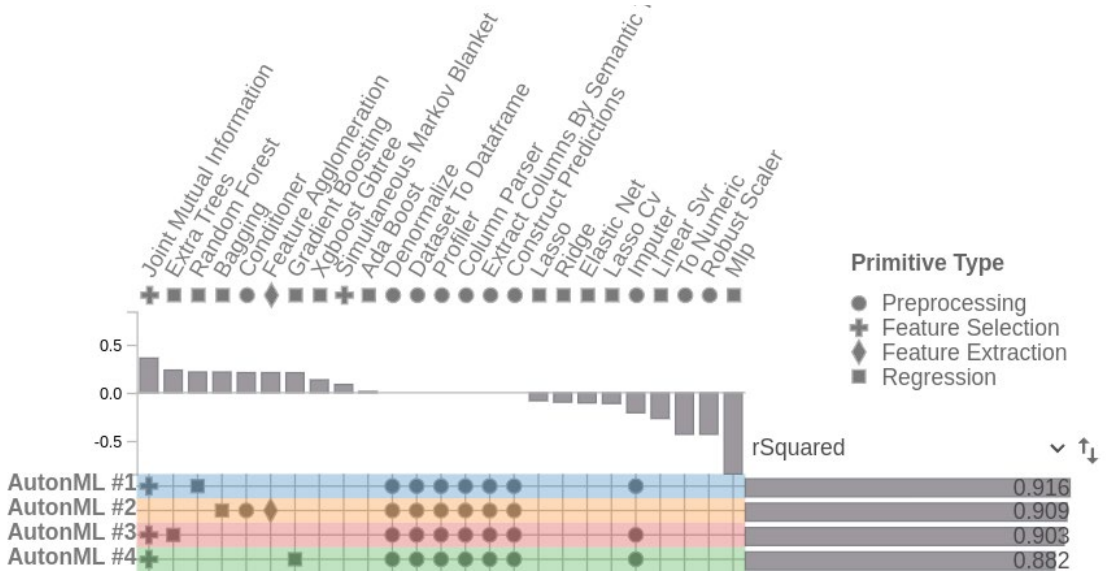

Figure 2: Example of Auto^{n}ML pipeline ranking of a regression task.

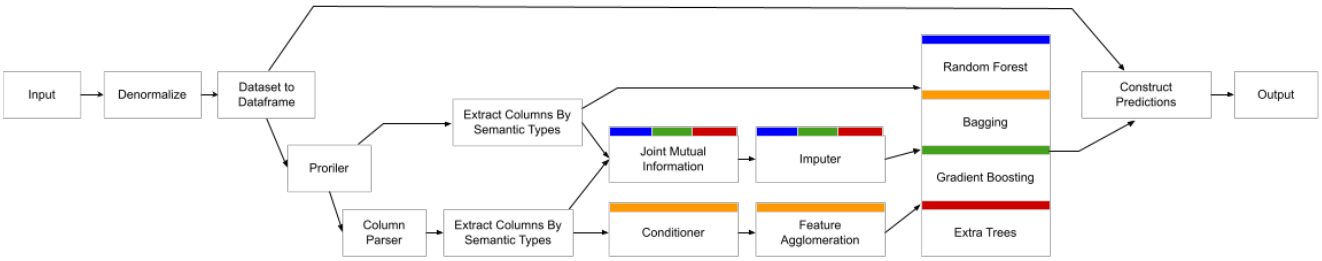

Figure 3: Illustration of Auto^{n}ML pipelines on a regression task.

Our D3M Auto^{n}ML software can be installed directly from the Gitlab repository and PyPI repository. It could run locally through Command Line Interface (CLI) and various Python IDEs. It could also be deployed in Podman containers, Docker, Amazon AWS platform and HPC environments with Jupyter Notebook interface.

Data scientists and domain subject matter experts (SME) could use Auto^{n}ML to identify designs with beneficial properties that can serve as alternatives to current solutions in order to validate their original design choices, quickly prototype solutions to new problems and estimate level of performance that can be attained in order to identify and qualify candidate problems for AI/ML. All these tasks have been prohibitively labor intensive creating a risk of missing good solutions to problems at hand, and of missing good opportunities for leveraging the power of AI/ML in practice. Auto^{n}ML mitigates those bottlenecks and scales up our productivity and effectiveness without the need for additional expert staff.

Our D3M Auto^{n}ML is configured to fit for integration with the TA2-TA3 API and is one of the

core AutoML engines in TA3 User Interface applications developed by our D3M collaborators including Enblink and Two Ravens.

4.2.2 Domain-Specific Applications

Example: Interdiction of radiation threat at border crossings

Enhanced Radiological and Nuclear Inspection and Evaluation (ERNIE) uses AI/ML for threat detection and characterization using radiation signatures measured on vehicles coming into the U.S. through ports of entry for the U.S. Customs and Border Protection (CBP), U.S. Department of Homeland Security (DHS).

In the ERNIE system, AI improves threat detection and significantly reduces the need for manual inspections of incoming cargo for potential radiological threat. We use Auto^{n}ML to verify if the current design of ERNIE AI is competitive versus newer model alternatives. Performance is evaluated at very low false positive rates ~0.1%. Auto^{n}ML evaluated on reference data with >50K data points and 135 features within 20 minutes. Competitive suites of threat classification pipelines were constructed, tested, and compared. The tool helps data scientists build optimal domain-specific models (including the use of semi-supervised learning).

Example: Image Classification - Stenosis Detection in Coronary Angiography

Segmentation and localization of potential blood vessel narrowings in images is one of the common data challenges for clinical diagnostics in coronary angiography. In predicting presence and severity of stenosis in coronary angiography images, Auto^{n}ML achieved performance on par with state of the art on a broader class of images than feasible ever before.

That image classification task is similar to target acquisition or anomaly detection applications of potential interest to United States Special Operations Command (SOCOM) and other Department of Defense (DoD) users, and also closely related to the RIMFIRE task undertaken by the Joint Artificial Intelligence Center (JAIC) as part of the JAIC Joint Logistics effort. It could be also extended to visual inspection of electrical wiring of Army UH-60 helicopters aimed to find irregularities such as wire chafing, loose plugs, etc. While human experts usually spend >1 day to train a feasible model, Auto^{n}ML model search time is only 174.23 seconds for the stenosis detection task.

Example: Bedside informatics in critical care

We used Auto^{n}ML to assess potential utility of a few newly defined predictive tasks on high density vital sign timeseries data from intensive care, and quickly validated attainable clinical impact of AI before investing significant development efforts.

Example: Predicting Engine Start Failures (UH-60 at 160th SOAR)

We used Auto^{n}ML to automate featurization in forecasting risk of over-temp on the Blackhawk helicopter engine startup. The flight data used in training is approximately 4TB and featurization is a big challenge given few target event examples in the data. We searched for featurization approaches and achieved false positive rate (FPR) < 4% at 15-25 flying hours forecast horizons with recall of future failures orders of magnitude better than current default. The results were briefed to the U.S. Senate Committee on Armed Services, have been integrated with the flight-line operations at the 160th Special Operations Air Regiment of the U.S. Army, and received comment of "In 2.5 years of use it has saved one squadron-year of workload".

4.2.3 Auto^{n}ML Performance Evaluation on OpenML Datasets

We evaluated Auto^{n}ML in comparison to existing open-source AutoML systems including H2O AutoML, Tree-Based Pipeline Optimization Tool (TPOT), AutoGluon and Auto-Sklearn using public OpenML datasets. The main goal was to assess the prediction performance of different AutoML systems on the same data tasks and identify the potential area of improvement of our Auto^{n}ML system. The current evaluation only focuses on end-results under different time constraints, instead of the implementation or pipeline search processes of different systems. The evaluation process has been automated in python scripts to download OpenML datasets (randomly chosen), convert into D3M format, run different AutoML systems and store all outputs.

Experiments over 177 classification tasks and 270 regression tasks were performed under the time budgets of 60, 600 and 1200 seconds. ROCAUC (Receiver Operating Characteristic/Area Under the Curve) scores and R-squared scores were used as evaluation metrics for classification and regression respectively. Rank statistics were used to quantify the performance of different AutoML tools on the same dataset. Though Auto^{n}ML did not beat TPOT and AutonGluon in classification tasks, it ranked top 1 on regression tasks under 600s and 1200s time budgets [7]. The average rank of AutonML shows an increasing trend when the dimensionality of datasets increases within the range of (0, 80) as shown in Figure 4 [8]. In repetitive experiments, AutonML shows much more consistent performance with comparison to TPOT as indicated in Figure 5 [8]. More detailed analysis of pipeline components are ongoing to continuously improve the tool.

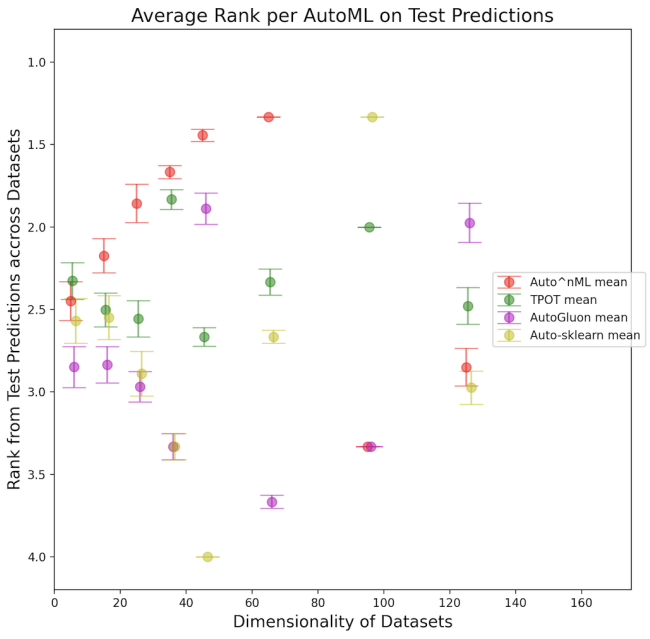

Figure 4: Average rank of AutoML systems with dimensionality of datasets for 270 benchmark regression tasks.

Figure 5: Performance comparison of AutonML and TPOT on OpenML dataset 491.

4.2.4 MILEI/Re-architectured AutonML

Though our D3M Auto^{n}ML achieved top performance in D3M internal evaluations, it also witnessed increasing issues in package dependencies for TA primitives such as KungfuAI version changes and python version upgrade from 3.8 to 3.9. Moreover, the native D3M framework falls short in the capability to integrate emerging large language models and pre-trained large models. For long-term opportunities and benefits of scalability and extensibility, we conducted the re-architecture effort of Auto^{n}ML with goals to remove dependencies to the D3M framework and substantially boost its configurability, flexibility, usability, and ultimately ease of adoption in practice.

Auto^{n}ML re-architecture is conducted with an emphasis on extensibility and usability for data scientist end users. It includes basic functionality for tabular classification and regression; it generates, runs, and ranks pipelines based on a problem definition provided by the user (which includes dataset, metadata, problem type, and metrics).

The new Auto^{n}ML system architecture is illustrated in Figure 6. The problem definition is a structured JSON file with user-specified problem type, data type, target output, data file path, metrics and other potential customizations including overriding of hyperparameters, list of models to include or exclude, cross-validation settings, etc. Pipeline template contains sequence of steps that are either a specific model or a model type (e.g. preprocessing, classifier, regressor, etc.). Templates are chosen based on data type and task type. Model catalog is a model registry for primitives, each of which has one or more flexible tags indicating usage (preprocessing, classifier, or regressor, etc.), software source (Sklearn, Auton Lab, etc.), customized tags for easy grouping used by pipeline templates. Model registration is also feasible to integrate pre-trained models. Metric catalog is a metric registry for different task types, designed with a customized metric registration process.

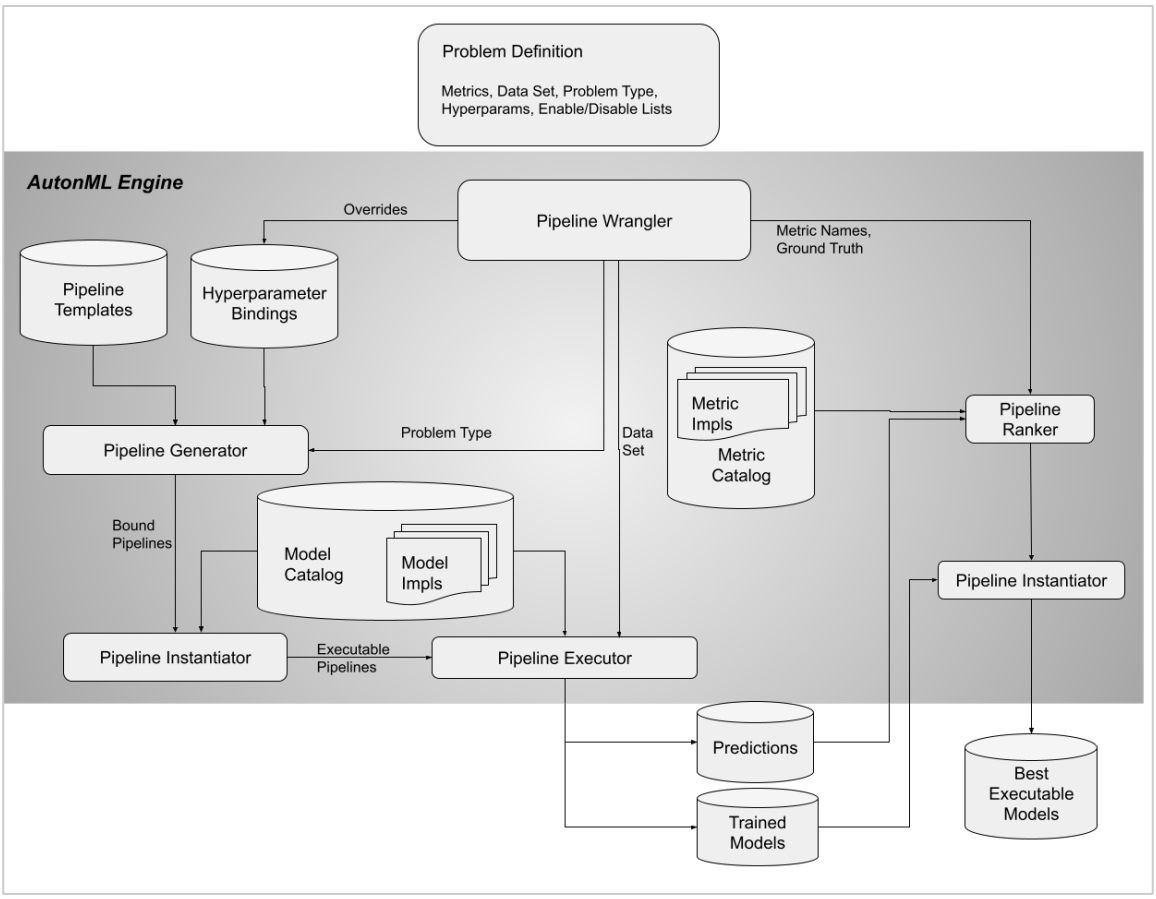

Figure 6: System Architecture of the new generation Auto^{n}ML (ngAuto^{n}ML).

We implemented agile practices (XP planning game, pair programming, continuous integration with test suite and linters, test-driven design) in support of the Auto^{n}ML re-architecture effort. We aimed for increased modularity and cohesion and decreased coupling as compared to the original Auto^{n}ML. Usage of succinct problem definitions has improved human-writeability: no need for a tool to transform 'regular' datasets into d3m datasets. We also quantified the effort needed to extend the system's menu with new models, new metrics, and new pipelines. Preliminary results suggest that new customized models can be added in as little as half an hour. Customized metrics are a little less, and new types of pipelines are more like a day. For existing scikit-learn packages, developers can add a new model or metric in 10 minutes.

With prioritization of ease of use, basic functionality, and robustness, we released the re-architectured version of the new generation Auto^{n}ML (ngAuto^{n}ML) in Q4 2023. ngAuto^{n}ML supports tabular datasets, image datasets and time series; integrates NeuralForecast NHiTS and NBEATSx; fully integrates Sklearn models and metrics for supported tasks and data types, while remaining open for further additions.

We also released a pip-installable version of new Auto^{n}ML tools with corresponding documentation. The PyPI version of ngAuto^{n}ML software can be found here https://pypi.org/project/ngautonml/ and its documentation: https://autonlab.gitlab.io/ngautonml/.

We designed a plugin structure in the architecture to allow different local versions of ngAuto^{n}ML tools with algorithms that are potentially incompatible with each other in dependencies, which will also benefit future integration of new algorithms and trained models. The plugin feature is only available in Gitlab repositories while the pip-installable version does not support the plugin feature yet.

5 CONCLUSIONS

The primary purpose of the CMU Auton Lab's Automated Machine Learning tool Auto^{n}ML is to multiply the capacity of Data Scientists by automating searches for plausible modeling process designs. It can help address shortages of qualified personnel and boost productivity of current staff. The research accomplishments made during the project have contributed to achieving the main goals of Auto^{n}ML. Meanwhile, we also experienced and tackled multiple challenges to continuously enhance the functionality of the framework and its software components and to maintain competitive advantages of Auto^{n}ML with emerging technologies to meet the evolving needs of its end users. Here is the list of key findings and lessons learned:

1. **Understanding the needs of different types of end users.** Data scientists and machine learning researchers are the major groups of end users of Auto^{n}ML at the software design and development phase. Understanding their analytic needs requires close engagement and interactions through end-user interviews for typical data analysis processes and end-result expectations, use case discussions, functionality testing and frequent feedback sessions. End users should be engaged at the design stages of the software to properly define the software development scope and guarantee relevance of the product. Similar process is important when new groups of end users, such as subject matter experts (SMEs) or non-technical users, are taken into consideration in developing dedicated user experiences (UXs).

2. **Productization of software tools through joint development collaboration and quick iterations.** As one of the key performers of the D3M project, we closely collaborated with different teams in joint software design, development, deployment, testing and evaluations with well-planned milestones to finally achieve the productization of the D3M framework and the Auto^{n}ML tool. Different versions of intermediate products were tested, evaluated, fixed and enhanced through quick iterations within the core D3M framework and also with our collaborators and their TA2 or TA3 software applications. D3M project and D3M community provided an open and friendly research environment for smooth communication and close collaboration. The productization of our Auto^{n}ML benefits from a wide range of assistance and inspirations of the D3M community including the Government Team and research collaborators.

3. **Data-driven evaluation of Auto^{n}ML performance with comparison to other AutoML tools.** A common question when engaging SMEs and new stakeholders is how our system performs versus other open-source or proprietary AutoML tools. Besides winning many evaluations within the D3M program, Auto^{n}ML demonstrates advantages in some classification and regression tasks when compared to external tools. Comprehensive data-driven evaluations should not be limited to overall performance ranking, but also in-depth diagnosis of data types

and ML pipeline nuances that appear to result in under-par performance. Detailed assessment approach helped us focus further development of the AutonML tool for continuous improvement.

4. **Importance of extensibility and scalability of the software architecture.** Our D3M Auto^{n}ML was challenged with increasing needs of end users to integrate new technologies, new computing environments and new customized processes. To be useful, our software architecture needs to be flexible and scalable for potential integration of APIs with new models such as new generations of foundational models. It also needs to be able to work across different computation environments (e.g., cloud environment, Linux/Windows/Mac, MLflow) and enable intelligent optimization for computing resources (CPU/GPU). The design of the architecture should not only consider user friendliness but also enable easy and fast development, debugging, and customizations.

We will continue our efforts in enhancing Auto^{n}ML to multiply the capacity of data scientists and encourage more end users to use our tool. Extensibility and scalability are critically important for Auto^{n}ML given the emergence of new technologies, new processes, and new computational environments.

6 REFERENCES

[1] Mitar Milutinovic, Brandon Schoenfeld, Diego Martinez-Garcia, Saswati Ray, Sujen Shah, David Yan, B Milutinovic, D Schoenfeld, S Martinez-Garcia, D Shah, Yan, Milutinovic, and Martinez-Garcia Ray Shah Schoenfeld. On evaluation of AutoML systems. 7th ICML Workshop on Automated Machine Learning, 2020.

[2] Biswajit Paria, Kirthevasan Kandasamy, and Barnabas Poczos. A flexible framework for Multi-Objective Bayesian Optimization using random scalarizations. Conference on Uncertainty in Artificial Intelligence (UAI), 2018.

[3] Biswajit Paria. Strategies for Black-Box and Multi-Objective Optimization. PhD thesis, 11 2022.

[4] Kin G. Olivares, Cristian Challu, Grzegorz Marcjasz, Rafal Weron, and Artur Dubrawski. Neural basis expansion analysis with exogenous variables: Forecasting electricity prices with NBEATSx. International Journal of Forecasting, 39(2):884–900, 2023.

[5] Cristian Challu, Kin G. Olivares, Boris N. Oreshkin, Federico Garza Ramirez, Max Mergenthaler Canseco, and Artur Dubrawski. NHiTs: Neural hierarchical interpolation for time series forecasting. Proceedings of the AAAI Conference on Artificial Intelligence, 37(6):6989–6997, June 2023

[6] Otilia Stretcu, Emmanouil Antonios Platanios, Tom Mitchell, and Barnabas Poczos. Coarse-to-Fine Curriculum Learning for Classification. In International Conference on Learning Representations (ICLR) Workshop on Bridging AI and Cognitive

Science (BAICS), 2020.

[7] Xinchen Yang, Jieshi Chen, and Artur Dubrawski. Evaluation of AutoML systems on OpenML binary- classification tasks. Carnegie Mellon Robotics Institute Summer Scholar Working Papers Journal, 10:256–269, 2022.

[8] Xinchen Yang, Jieshi Chen, and Artur Dubrawski. Evaluation of AutoML Systems On OpenML Regression Tasks. Carnegie Mellon Robotics Institute Summer Scholar Working Papers Journal, 2023. (to be published).

APPENDIX A - Publications and Presentations

[CC22] Cristian Challu, Peihong Jiang, Ying Nian Wu, and Laurent Callot. Deep generative model with hierarchical latent factors for time series anomaly detection. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, Proceedings of The 25th International Conference on Artificial Intelligence and Statistics (AISTATS), March 2022.

[KK19] Kirthevasan Kandasamy. Tuning Hyperparameters without Grad Students: Scaling up Bandit Optimisation. PhD thesis, 7 2019.

[CN19] Chirag Nagpal, Xinyu Li, Michael R Pinsky, and Artur Dubrawski. Dynamically personalized detection of hemorrhage. In Machine Learning for Healthcare Conference (MLHC), pages 109–123. PMLR, 2019.

[MS21] Maya Sitaram, Jieshi Chen, and Artur Dubrawski. Evaluation of AutoML systems using OpenML datasets. Carnegie Mellon Robotics Institute Summer Scholar Working Papers Journal, 9:274–283, 2021.

[GS20] George Stoica, Otilia Stretcu, Emmanouil Antonios Platanios, Tom Mitchell, and Barnab as P oczos. Contextual parameter generation for knowledge graph link prediction. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 3000–3008, 2020.

[OS20] [YX21] Yichong Xu. Learning and Decision Making from Diverse Forms of Information. PhD thesis, Carnegie Mellon University, 2021.

[CL17] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabas Poczos. Mmd gan: Towards deeper understanding of moment matching network. In Neural Information Processing Systems (NeurIPS), 2017.

[KK16] Kirthevasan Kandasamy, Gautam Dasarathy, Junier B Oliva, Jeff Schneider, and Barnabas Poczos. Gaussian process bandit optimization with multi-fidelity evaluations. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, Advances in Neural Information Processing Systems (NeurIPS), 2016.

[KK17] Kirthevasan Kandasamy, Gautam Dasarathy, Jeff Schneider, and Barnabas Poczos. Multi-fidelity Bayesian optimization with continuous approximations. In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning (ICML), 2017.

[KJ17] Kirthevasan Kandasamy, Jeff Schneider, and Barnabas Poczos. Query efficient posterior estimation in scientific experiments via Bayesian active learning, Artificial Intelligence 243(C):45–56, February 2017.

[KK18] Kirthevasan Kandasamy, Willie Neiswanger, Jeff Schneider, Barnabas Poczos, and Eric P. Xing. Neural architecture search with bayesian optimisation and optimal transport. In Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS), 2018.

[KA18] Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabas Poczos. Parallelised Bayesian optimisation via Thompson sampling. In Amos Storkey and Fernando Perez-Cruz, editors, Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics (AISTATS), April 2018.

[KK19] Kirthevasan Kandasamy, Willie Neiswanger, Reed Zhang, Akshay Krishnamurthy, Jeff Schneider, and Barnabas Poczos. Myopic posterior sampling for adaptive goal oriented design of experiments. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, Proceedings of the 36th International Conference on Machine Learning (ICML), June 2019.

[IA19] Ifigeneia Apostolopoulou, Scott Linderman, Kyle Miller, and Artur Dubrawski. Mutually regressive point processes. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alch é-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems (NeurIPS), 2019.

[AU19] Ananya Uppal, Shashank Singh, and Barnabas Poczos. Nonparametric density estimation & convergence rates for GANs under Besov IPM losses. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alch é-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems (NeurIPS) Inc., 2019.

[SS18] Shashank Singh, Ananya Uppal, Boyue Li, Chun-Liang Li, Manzil Zaheer, and Barnabas Poczos. Nonparametric density estimation under adversarial losses. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, Advances in Neural Information Processing Systems (NeurIPS), 2018.

[YX18] Yichong Xu, Hariank Muthakana, Sivaraman Balakrishnan, Aarti Singh, and Artur Dubrawski. Nonparametric regression with comparisons: Escaping the curse of dimensionality with ordinal information. In Jennifer Dy and Andreas Krause, editors, Proceedings of the 35th International Conference on Machine Learning (ICML), July 2018.

[YW19] Yuexin Wu, Yichong Xu, Aarti Singh, Yiming Yang, and Artur Dubrawski. Active learning for graph neural networks via node feature propagation. CoRR, abs/1910.07567, 2019.

[MB19] Matt Barnes and Artur Dubrawski. On the interaction effects between prediction and clustering. In Kamalika Chaudhuri and Masashi Sugiyama, editors, The 22nd International Conference on Artificial Intelligence and Statistics, (AISTATS), 2019.

[BB19] Benedikt Boecking and Artur Dubrawski. Pairwise feedback for data programming. In Proceedings of NeurIPS '19 Workshop on Learning with Rich Experience (LIRE '19), December 2019.

[MD19] M. De-Arteaga, J. Chen, P. Huggins, J. Elmer, G. Clermont, and A. Dubrawski. Predicting neurological recovery with Canonical Autocorrelation Embeddings. PLoS One, 14(1):e0210966, 2019.

[DB19] David Bethge, Jieshi Chen, Oliver Grothe, Jonathan Elmer, and Artur Dubrawski. Prognostication of neurological recovery by analyzing structural breaks in eeg data. In 2019 International Conference on Data Mining Workshops (ICDMW), pages 933–940, 2019.

[NG18] Nick Gisolfi and Artur Dubrawski. Revealing actionable simplicity in data. In Proceedings of AAAI '18 Spring Symposium on Design of User Experience for Artificial Intelligence, pages 382 – 385, March 2018.

[LC19] Lujie Chen, Artur Dubrawski, Gilles Clermont, Tiffany Pellathy, Anthony Wertz, Joo Heung Yoon, Michael Pinsky, and Marilyn Hravnak. Binarized severity level of future instability risk in continuously monitored patients. Critical Care Medicine, 47(1):605, January 2019.

[CG] Chufan Gao, Fabian Falck, Mononito Goswami, Anthony Wertz, Michael R. Pinsky, and Artur Dubrawski. Detecting patterns of physiological response to hemodynamic stress via unsupervised deep learning, NeurIPS 2019 ML for Healthcare (ML4H) Workshop, 2019.

[JY20] J. H. Yoon, V. Jeanselme, A. Dubrawski, M. Hravnak, M. R. Pinsky, and G. Clermont. Prediction of hypotension events with physiologic vital sign signatures in the intensive care unit. Critical Care Medicine, 24(1):661, November 2020.

[WC19] Wei-Cheng Chang, Chun-Liang Li, Yiming Yang, and Barnabas Poczos. Kernel change-point detection with auxiliary deep generative models, International Conference on Learning Representations (ICLR), 2019.

[ZS17] Zoltan Szabo, Bharath K. Sriperumbudur, Barnabas Poczos, and Arthur Gretton. Learning theory for distribution regression. Journal of Machine Learning Research, 17(1):5272–5311, January 2016.

[XL18] Xinyu Li, Michael R. Pinsky, Gilles Clermont, and Artur Dubrawski. Leveraging routine pre-operative blood draws to predict hemorrhagic shock during surgery. In Proceedings of NeurIPS 2018 Workshop on Machine Learning for Healthcare (ML4H), December 2018.

[JO18] Junier Oliva, Avinava Dubey, Manzil Zaheer, Barnabas Poczos, Ruslan Salakhutdinov, Eric Xing, and Jeff Schneider. Transformation autoregressive networks. In Jennifer Dy and Andreas Krause, editors, Proceedings of the 35th International Conference on Machine Learning (ICML), July 2018.

[YS18] Yichong Xu, Sivaraman Balakrishnan, Aarti Singh, and Artur Dubrawski. Interactive linear regression with pairwise comparisons. In 2018 52nd Asilomar Conference on Signals, Systems, and Computers, pages 636–640, 2018.

LIST OF ACRONYMS

CBP – Customs and Border Protection
CLI – Command Line Interface
CMU – Carnegie Mellon University
D3M – Data-Driven Discovery of Models
DHS – Department of Homeland Security
DoD – Department of Defense
ERNIE – Enhanced Radiological and Nuclear Inspection and Evaluation
ES-RNN – Exponential Smoothing-Recurrent Neural Networks
HPC – High-Performance Computing
IDE – Integrated Development Environment
JAIC – Joint Artificial Intelligence Center
ML – Machine Learning
NBEATSx – Neural Basis Expansion Analysis with exogenous variables
ngAuto^{n}ML – New Generation Auto^{n}ML
NHiTS – Neural Hierarchical Interpolation for Time Series
PyPI – Python Package Index
SME – Subject Matter Expert
SOCOM – United States Special Operations Command
SOTA – Special Operations Team-Alpha
TA1/TA2 – Technical Area 1, 2
TFT – Temporal Fusion Transformer
TPOT – Tree-Based Pipeline Optimization Tool