Applied Computer Systems
ISSN 2255-8691 (online)
ISSN 2255-8683 (print)
December 2024, vol. 29, no. 2, pp. 43–52
https://doi.org/10.2478/acss-2024-0020
https://content.sciendo.com

A Comparative Analysis of Automated Machine Learning Libraries for Electricity Price Forecasting

Christian O'Leary[1*], Conor Lynch[2], Farshad Ghassemi Toosi[3]
[1,3]*Department of Computer Science, Munster Technological University, Cork, Ireland*
[2]*Nimbus Research Centre, Munster Technological University, Cork, Ireland*

***Abstract* – Reliable and accurate electricity price forecasting algorithms can be used to inform efficient energy consumption schedules and maximise profits for electricity traders. Operating within Ireland's Integrated Single Electricity Market (I-SEM), traders can buy and sell electricity at fluctuating hourly rates whose day-ahead prices are published at approximately 13:00 GMT day-1. Access to electricity price predictions earlier than this publication time allows stakeholders an expanded timeframe to facilitate energy cost-aware scheduling.**

While many studies have been conducted to espouse various machine learning and statistical approaches to electricity price forecasting, these models tend to be bespoke and require in-depth knowledge regarding model implementation. The problem of requiring such expertise is not unique to time series forecasting, and research into mitigating such limitations exists in the form of Automated Machine Learning (AutoML). AutoML aims to derive effective models while automating various steps typically required for machine learning experimentation, such as pre-processing, model selection, validation, etc.

Given the increasing proliferation of AutoML tools and frameworks, this paper applies eight Python-based AutoML libraries to day-ahead electricity price forecasting on an excerpt of I-SEM data. These libraries are compared across a series of error metrics and training times to produce an empirical benchmark that can be utilised to select high-performing AutoML tools for further price forecasting research and other forms of time series forecasting. AutoKeras is found to produce accurate forecasts but requires careful configuration to avoid long runtimes. PyCaret, Ludwig, FLAML and FEDOT also generate favourable results while being significantly easier to configure.

***Keywords* – Artificial intelligence, deep learning, forecasting, machine learning.**

I. Introduction

Ireland's electricity infrastructure transitioned from the Single Electricity Market (SEM) to the Integrated SEM (I-SEM) in October 2018 [1]. The I-SEM is an all-island market to facilitate electricity exchange, cross-border transmission, and dynamic price calculation. A joint venture between EirGrid plc and SONI Limited, the Single Electricity Market Operator (SEMO) administrates the I-SEM through the national power exchange SEMOpx [2]. Here, energy stakeholders can buy/sell electricity through fluctuating day-ahead hourly prices, which are published daily. Electricity prices affect the operating efficiency and profit margins of market participants [3], motivating the need for accurate predictions of these prices.

The provision for reliable electricity price forecasts is invaluable to both electricity consumers and traders given its pervasiveness in all sectors of society and industry [4]. As a result, researchers have applied a variety of modelling techniques to both Irish [5]–[8] and other international markets [4], [9]–[11]. State-of-the-art approaches tend to focus on statistical and machine learning algorithms over other methods such as multi-agent systems or econometric models [4]. Traditional approaches such as system simulation necessitate complex implementation and require expert domain knowledge [12]. Modelling electricity prices is uniquely challenging due to volatility, anomalies [13] and geopolitical tensions. With decentralised renewable energy sources becoming increasingly common, the correlation between fossil fuel prices and electricity tariffs has waned [14]. Consequently, the evolving complexity of electricity markets necessitates more sophisticated modelling approaches.

While some works emphasise the efficacy of machine learning models over traditional statistical methodologies for time series forecasting [9], others have illustrated that statistical models are competitive in terms of prediction error metrics for data derived from the I-SEM [5]. Statistical model outputs can also be used as machine learning model inputs to increase overall prediction accuracy [4]. Machine learning models employed in the prediction of electricity prices include support vector machines [15], artificial neural networks[2] (ANNs) with Fuzzy inference [16], deep learning neural networks such as Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models [17], decision trees, K-Nearest Neighbours, and linear models such as ElasticNet [18].

The implementation of machine learning models continues to be a challenging prospect for many aspiring researchers due to the high knowledge and skill thresholds required to implement the various required steps such as pre-processing, model development, hyper-parameter tuning, etc. [19]. Mathematical

* Corresponding author's e-mail: christian.oleary@mtu.ie
Article received 2024-04-24; accepted 2024-10-11
[2] The term 'ANN' is most typically used to describe smaller densely connected neural networks such as multi-layer perceptrons but is sometimes used interchangeably with NN in academic literature.

time series models such as Holt-Winters can also be time-consuming to implement even with a working knowledge of the algorithm [3]. Given the difficulties in designing and programming time series forecasting models, automated machine learning (AutoML) methodologies have seen increasing traction in academic research [4]. AutoML aims to simplify the process of optimising models by automating as many of the associated imperative tasks as possible, e.g., scaling, imputation, data augmentation, model selection, hyperparameter optimisation, model validation, serialisation, etc. [20]. Unfortunately, most AutoML research has been directed towards other domains such as image classification, which mirrors the trends observable in machine learning research in general [21]. Time series forecasting has comparatively been the subject of fewer works, necessitating more research on the subject [22]. AutoML implementations use complex algorithms to automatically optimise machine learning pipelines such as neural architecture search (NAS), evolutionary algorithms (genetic algorithms, particle swarm optimisation), reinforcement learning (RL) and Bayesian search approaches.

The development of forecasting methodologies using the aforementioned algorithms can be executed using programming languages such as Python, R and MATLAB. Python is by far the most widely used language of the three and was the third most used programming language overall in a 2023 Stack Overflow survey of 87 585 programmers being surpassed only by JavaScript and 'HTML/CSS' [23]. The Python Package Index (PyPi) provides access to a plethora of libraries for its users such as NumPy and Pandas (data manipulation), scikit-learn (machine learning), TensorFlow (deep learning) and statsmodels (statistical and time series models). These libraries allow researchers and engineers to design, train and test models using a computer's CPU and GPU[3] where applicable. Previous research works [24] have presented empirical results using such libraries to perform I-SEM electricity price forecasting. However, none have examined the efficacy of existing AutoML libraries and frameworks. The work presented in this paper aims to address this shortcoming in the state-of-the-art by examining the viability of applying AutoML algorithms to I-SEM day-ahead electricity price forecasting. Furthermore, an empirical analysis is presented to rank the effectiveness and efficiency of various AutoML libraries using forecasting error metrics and runtime comparisons. This benchmark of experimental data is intended to inform prospective AutoML programmers on the efficacy of the tested AutoML libraries for electricity price forecasting. The methodology is, however, not domain-dependent, and can be adapted to other domains with relative ease.

Section I provides an overview, motivation and contribution of this research. Section II informs the reader of existing literature and related research pertinent to I-SEM electricity price forecasting. Section III describes the applied methodology of this explorative analysis, including the AutoML libraries and experiment design. Section IV presents an evaluation of resulting AutoML library performances using error, ranking and runtime metrics. Finally, Section V summarises the findings of the research and proposes avenues for future research.

II. Background

Previous research into electricity price forecasting in Irish electricity markets (SEM and I-SEM) predominantly espouses the use of machine learning models and statistical models over other methodologies such as multi-agent simulation and physical simulation. Using data from 2016 and 2017 respectively, Li et al. [25] presented an ANN model utilising an adaptive neural network-based fuzzy inference system (ANFIS) to provide electricity price forecasts for the now obsolete SEM arrangement. In 2018, Arci et al. [14] expanded this methodology to incorporate hyperparameter optimisation algorithms and Recursive Feature Engineering (RFE) using a Random Forest intermediate model for selecting optimal features. The resulting ANN models were found to produce favourable Mean Absolute Error (MAE) scores between 12.7 and 19.04[4].

Using SEM data, Grimes et al. [8] demonstrated the application of two SVM forecasting models (FM) – designated FM1 and FM2 – in day-ahead price forecasting. Both SVM models reported lower Mean Squared Error (MSE) and MAE scores than official hourly day-ahead SEMO €/MWh price predictions. Schedules were generated using Mixed Integer Linear Programming (MILP) for two simulated scheduling problems using the three forecasts (SEMO, FM1, and FM2). Although FM2 achieved the lowest MSE score (781.72), its predictions resulted in the worst (most expensive) schedules, illustrating that using simple metrics such as MAE and MSE can yield misleading results, as they do not measure price ranking. As a means of mitigation, Grimes et al. presented three bespoke metrics for model selection prior to schedule generation, with the Geometric mean of Spearman Rank Correlation (SRC) and MAE being a more effective measure of a forecasting model's suitability for scheduling applications while being trivial to compute. This new metric was used to optimise the FM1 model to produce schedules costing only 2.92 % more than using a perfect forecast (FM2 was 3.28 % worse). The other two custom metrics provided comparable results, but their calculations were use-case-dependent.

Lynch et al. [24] applied a hybrid K-Means-SVM-SVR model (K-SVM-SVR or K-SVR) to electricity price forecasting in the SEM achieving an MSE of 712.89 versus 781.72 from [8] and 1086.25 for the official SEMO forecast, representing a 34.37 % improvement. Based on a similar methodology to [15], a K-Means algorithm was used to separate the data into K clusters with which K SVR (Support Vector Regression) models were trained. The labels generated by the K-Means model were used to train an SVM classifier to select the appropriate SVR model during inference. Recursive forecasting

[3] Via the CUDA toolkit: https://developer.nvidia.com/cuda-toolkit

[4] It is assumed that the "Actual Error" referred to in the paper corresponds to MAE. Both papers erroneously refer to the target variable as "load". However, the models are trained to forecast System Marginal Price (SMP) values.

was used to provide a 24-hour forecast horizon, i.e., model outputs for $hour_{t=1}$ could be used as features for predicting values at $hour_{t=2}$, etc. An expansion of this set of results culminated in a follow-on work that reviewed a series of machine learning models on data from the then newly established I-SEM [18]. Operating on a dataset comprising 438 days (30 Sep. 2018 to 12 Dec. 2019), thirteen machine learning models and twenty-four deep learning architectures were optimised to provide day-ahead electricity price forecasts using the aforementioned recursive methodology. While some machine learning models such as K-Nearest Neighbours and Random Forest were discovered to result in lower error scores (11.21 and 11.54, respectively) than the K-SVMSVR model (11.97), the deep learning models failed to match this performance while requiring far longer training times. It was proposed that a shortcoming of the work was that the feature selection methods (feature score curves) were susceptible to human error. In 2021, Lynch et al. [6] demonstrated that expanding the prospective feature set to incorporate wind speeds and gas prices could improve I-SEM day-ahead price forecasting performance; a Light Gradient Boosting Machine (LGBM) and an Extreme Gradient Boosting Machine (XGBM) produced MAE scores of 9.58 and 10.08, respectively.

While machine learning approaches are seen as popular and effective algorithms in the state of the art, McHugh et al. [5] exemplified the efficacy of a statistical approach via Nonlinear AutoRegressive Moving Average model with eXogenous inputs (NARMAX) methodology paired with a Seasonal AutoRegressive Integrated Moving Average model with eXogenous input (SARIMAX). The SARIMAX model was fitted using features selected from factors such as historical prices, demand and CO2 using AutoCorrelation Function (ACF) and Partial ACF (PACF) plots. The resulting RMSE scores ranged from 13.99 to 15.15. The lowest RMSE of [6] was 15.86, although a direct comparison is difficult as the experiments of both papers used different samples of data. The result is nonetheless significant as many machine learning models (especially neural networks) are more computationally complex and are, therefore, slower to train.

To the best of the authors' knowledge, there have not been any studies applying AutoML approaches to Irish I-SEM data. Beltran et al. [4] claim to be the first published study presenting an AutoML methodology applied to electricity price forecasting in the Spanish market. The paper describes an advanced amalgamation of feature engineering and selection techniques, model optimisation and selection, and the inference of time series components such as trend, seasonality (12, 24, 84, and 168 hours) and the remainder component. The complexity of implementing such as workflow may contribute to the lack of AutoML publications in the area of electricity price forecasting. As a result, the research presented in this paper is envisioned to provide a foundation for research within the topic, with a particular focus given to forecasting models based on existing AutoML libraries that are publicly accessible and may be more easily incorporated into research or development than bespoke algorithms.

III. Methodology

The methodology outlined in this paper is intended to thoroughly evaluate the performance of AutoML libraries with regard to day-ahead electricity price forecasting in the I-SEM. With accessibility and reproducibility in mind, open-source libraries with high-level time series-specific classes or entry points are favoured. Python-based libraries are selected for reasons explained in Section I.

To the authors' knowledge, the 01/01/2020 to 31/12/2020 I-SEM dataset used previously in [6] represents the most recent dataset used in I-SEM day-ahead electricity price forecasting studies. Conveniently, it also represents one calendar year which aids reproducibility for any future related works. Given the long runtimes involved in training machine learning models using the shortlisted AutoML libraries, univariate data has been selected as opposed to multivariate data, i.e., the libraries only use historical electricity prices to create feature sets from which they perform univariate or multivariate forecasting depending on their implementation.

As mentioned previously, only Python-based AutoML libraries with time series-specific classes or functions are included in this work resulting in a shortlist of eight popular open-source libraries: *AutoGluon* [26], *AutoKeras* [27], *AutoTS* [28], *ETNA* [29], *FEDOT* [30], *Fast and Lightweight AutoML Library (FLAML)* [31], *PyCaret* [32], and *Ludwig* [33]. The versions used for each library are recorded in the Appendix. As presented in a previous study by O'Leary et al. [20], there are a plethora of other AutoML libraries that do not have time series-specific functions or classes such as auto-sklearn [34], LightAutoML [35], and MLBox [36]. Frameworks such as H2O [37] do have time series functionality, but such functionality is not available as an AutoML module. EvalML [38] is excluded from this study due to a technical limitation, i.e., it assumes that test sets are the same length as the forecast horizon[5] and cannot perform rolling origin forecasting without retraining, which is typical of electricity price forecasting and scheduling simulations.

A quantitative overview of the functionalities of each of the eight selected libraries is provided in [20] and details of their available models are outlined in the Appendix. *AutoGluon* [26] is a general-purpose AutoML library with capabilities for statistical, machine learning, deep learning, and ensemble models with time series presets `best_quality`, `high_quality`, `medium_quality` and `fast_training`. *AutoKeras* [27] is built on the Keras API of TensorFlow using KerasTuner to automatically optimise deep learning models using Greedy, Random, Bayesian, and Hyperband optimisation algorithms. *AutoTS* [28] is a time series-specific library that constructs machine learning, deep learning and statistical models using Genetic Algorithms (GA).

[5] https://evalml.alteryx.com/en/stable/user guide/timeseries.html

TABLE I
MAIN AUTOML LIBRARY PARAMETERS

Library	Parameter	Values
AutoGluon	`time_limit`	Integer in seconds
	`presets`	`best_quality`, `high_quality`, `medium_quality`, `fast_training`
AutoKeras	`epochs`*	Integer: 10, 50, 100, 150
	`tuner`	`greedy`, `bayesian`, `hyperband`, `random`
AutoTS	`max_generations`	Calculated as the time limit divided by the `generation_timeout`
	`generation_timeout`*	Integer: 1, 2, 4, 8, 16, 32 minutes
	`model_list`	`superfast`, `fast`, `fast_parallel`, `default`, `all`
ETNA	`tune_size`**	Integers sampled from 1–500
	`n_trials`**	Integers sampled from 1–500
FEDOT	`timeout`	Integer in minutes
	`preset`	`fast_train`, `ts`, `gpu`, `stable`, `best_quality`, `auto`
FLAML	`time_budget`	Integer in seconds
Ludwig	`config.epochs`*	Integer: 10, 50, 100, 150, 200
PyCaret	`budget_time`	Integer in seconds

* Integer values (excluding time limits) were selected using the specified ranges via Grid Search to ensure a fair comparison.

** All configurations for ETNA resulted in the same model output. The meanings of the categorical preset parameters for AutoGluon, AutoTS and FEDOT are explained in detail in the Appendix.

AutoTS achieved the first place in the Decisions category of the M6 Forecasting Competition [39]. *ETNA* [29] is a library specifically designed for time series analysis. ETNA optimises pipelines using Optuna, which implements a Tree-structured Parzen Estimator optimiser. *FEDOT* [30] automates ML pipeline development using graph-based metaheuristics such as GAs and Neural Multi-Armed Bandits. The Fast and Lightweight AutoML Library (*FLAML*) [31] is an AutoML library that optimises models using BlendSearch and CostFrugal Optimisation via a simple API. *PyCaret* [32] is a library for generic machine learning modelling with an entry point for time series forecasting. PyCaret requires relatively little configuration to run. *Ludwig* [33] is a multi-domain library with a lower-level API that can operate with simple configurations but also supports more complex configurations via a generic `config` parameter.

A. Implementation and Workflow

The aforementioned AutoML libraries are provided with 293 days (~80 %) of the available data for training while 73 days (~20 %) are reserved as a test set for rolling origin forecasting. Each 'day' consists of 24 hourly values corresponding to the Republic of Ireland Day-Ahead (*ROI-DA*) I-SEM market. Models are expected to produce a forecast of 24 values *simultaneously* to represent the incoming trading day prices. Each library is trained using a time limit of one hour where overall time limits are enforceable via input parameters. Shorter training times are desirable to facilitate forecast generation, schedule formulation and schedule application, which happen on a daily basis in the day-ahead market. At the time of implementation, four of the eight examined libraries expose a time limit parameter: AutoGluon, FEDOT, FLAML and PyCaret. An attempt to cap the runtimes of AutoKeras, AutoTS, ETNA and Ludwig was implemented indirectly through the configuration of alternative parameters, although this was superfluous for Ludwig and ETNA which consistently executed within the time constraint. AutoKeras[6], AutoTS and ETNA allow users to specify trial limits for their underlying optimisers. AutoTS additionally provides a time limit `generation_timeout` for GA generations although these limits are not strictly adhered to. ETNA implements a `tune_size` parameter which determines how many pipelines to tune. AutoKeras and Ludwig include a parameter for maximum epochs allowed during neural network training.

The libraries expose unique APIs; this research attempted to leave as many of these parameters at their default values as possible without introducing bias into the results by limiting certain libraries to inappropriate configurations. Crucially, some of the AutoML libraries have parameters that dramatically influence the efficacy of library invocation by setting limits on which model types can be used, as outlined in Table I. The meanings of the categorical preset parameters for AutoGluon, AutoTS and FEDOT are explained in detail in the Appendix. AutoGluon's `time_limit`, FEDOT's timeout, FLAML's `time_budget` and PyCaret's `budget_time` refer to overall model training time constraints. The `config.epochs` parameter of Ludwig is a Python dictionary; the documentation of this parameter for the time series API is unfortunately limited.

The Python code used to generate results for this benchmark is available online[7] to aid the reproducibility and interpretability of the work. As the libraries have many conflicting dependencies, each is executed in a separate Python 3.9

[6] AutoKeras previously had a time limit parameter, but this was removed: https://github.com/keras-team/AutoKeras/issues/920

[7] https://github.com/christian-oleary/AutoML-Python-Benchmark/tree/electricity_price_forecasting

environment in Windows 10 using CUDA 11.2 and cudnn 8.1. TensorFlow 2.10.0 and PyTorch 2.0.1 are used as default versions; these are altered for specific libraries if their installation and operation require an alternative version. The latest library version was used for each AutoML library unless internal bugs prevented execution. The experiments were run on an Intel Core i910980XE CPU with up to 256 GB of RAM and an NVIDIA RTX 6000 Ada GPU; these device specifications exceed what is typically required for training models on a dataset of this size. This is a deliberate design choice to ensure that any computational bottlenecks or inefficiencies are not introduced by the device's hardware configuration and are instead attributable to the AutoML libraries under examination. This is intended to ensure an unbiased comparison between tools.

Some libraries including AutoGluon required minor data reformatting before training using library functions such as `TimeSeriesDataFrame.from_data_frame`, `to_regular_index` and `fill_missing_values`, whilst others, including AutoKeras, AutoTS and FEDOT, required data to be reformatted into a tabular format prior to fitting. This was facilitated using bespoke code [8] that prepared a tabular dataset by creating features using shift operations from lagged historical values, preparing target or *y* columns, and replacing the resulting missing values using scikit-learn's IterativeImputer class.

To provide a baseline comparison with the examined AutoML libraries, four additional forecasting models were implemented. A Multiple Linear Regression (MLR) model along with the LGBM and XGBM models espoused in [6] were each fitted using the aforementioned engineered tabular dataset and are referred to in the overall results of the paper. Additionally, a Naïve model was implemented by forecasting the electricity price at time *t* using the value from time *t*-24.

B. Metrics

As described in Section II, some popular electricity price forecasting metrics include MAE, MSE and RMSE, which are explained by (1), (2) and (3), respectively. Additionally, the Mean Absolute Scaled Error (MASE) scores are included due to scale insensitivity making them useful for general comparisons [40]. The formula for MASE is given by (4). However, as explained by Grimes et al. [8], metrics based on forecast error alone do not account for value ranking, which is a critical component for scheduling applications. Therefore, SRC (5) values are reported in this study along with the geometric mean of MAE and SRC (GM-MAE-SRC). The method described by Grimes et al. [8] is to maximise the geometric mean of negative MAE (as MAE is an error metric) and SRC. As geometric means cannot be calculated with negative values and it is unclear how negative MAE values were interpreted in [8] (e.g., by treating them as percentage values), the formula used in this study instead minimises the geometric mean of MAE and (1-SRC), which is facilitated by the fact that SRC cannot be greater than 1.

$$\mathrm{MAE}(y,\hat{y}) = \frac{\sum_{i=0}^{N-1}|y_i-\hat{y}_i|}{N}; \quad (1)$$

$$\mathrm{MSE}(y,\hat{y}) = \frac{\sum_{i=0}^{N-1}(y_i-\hat{y}_i)^2}{N}; \quad (2)$$

$$\mathrm{RMSE}(y,\hat{y}) = \sqrt{\frac{\sum_{i=0}^{N-1}(y_i-\hat{y}_i)^2}{N}}; \quad (3)$$

$$\mathrm{MASE}(y,\hat{y}) = \frac{\frac{1}{N}\sum_{i=0}^{N-1}|y_i-\hat{y}_i|}{\frac{1}{N-1}\sum_{i=1}^{N-1}|y_i-y_{i-1}|}; \quad (4)$$

$$\mathrm{SRC}: \rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)}; \quad (5)$$

$$\text{GM-MAE-SR} = \sqrt{\mathrm{MAE}(1-\mathrm{SR}).} \quad (6)$$

IV. Evaluation

To examine the relationship between the available metrics, a Pearson Correlation matrix was plotted and displayed in Table II. All p-values are less than 0.05. SRC is not normally distributed, which violates Pearson correlation analysis prerequisites and is omitted accordingly. Perhaps, unsurprisingly, the error metrics MAE, MASE, MSE, and RMSE correlate with each other with values ranging from 0.71 ($p = 4.32e^{-15}$) to 0.98 ($p = 3.84e^{-81}$). Interestingly, however, the correlations between these four error metrics and GM-MAE-SR range from 0.4 ($p = 0.0006$) to 0.58 ($p = 2.48e^{-14}$). This discrepancy mirrors the findings of [8], i.e., error metrics alone are limited in predicting model performance, where ranking is required for scheduling applications, e.g., electricity price forecasting.

TABLE II
Pearson Correlation Matrix of Forecasting Metrics

	GM-MAE-SR	MAE	MASE	MSE	RMSE
GM-MAE-SR	1	0.46	0.38	0.59	0.58
MAE	0.46	1	0.87	0.93	0.87
MASE	0.38	0.87	1	0.71	0.56
MSE	0.59	0.93	0.71	1	0.96
RMSE	0.58	0.87	0.56	0.96	1

The results of running each library iteratively over the available presets are summarised in Table III. For brevity, only the top 5 performing presets with respect to GM-MAE-SR of AutoKeras and AutoTS are presented. Additionally, one additional result for AutoKeras is included that obeys the 1-hour time limit to facilitate a fairer comparison as the top five results exceed the 1-hour time constraint.

[8] https://github.com/christian-oleary/AutoML-Python-Benchmark/blob/c436f3f83e6872ab8a4bb430923fc5aaf64f5ade/src/base.py#L176

TABLE III
FORECASTING ERROR SCORES AND COMPUTATION TIMES ORDERED BY GM-MAE-SR

Library / Model	Preset	Duration (seconds)	GM-MAE-SR ↓	MAE	MASE	MSE	RMSE	SRC
AutoKeras	60-50-Random**	23 162.51	1.91	15.36	3.30	668.40	25.26	0.75
	60-50-Greedy**	20 077.88	1.92	15.21	3.27	644.21	24.94	0.75
	60-100-Greedy**	31 009.08	1.95	16.09	3.46	643.94	25.08	0.75
	60-10-Random**	6226.28	2.04	17.57	3.77	879.80	29.62	0.74
	60-100-Bayesian**	30 349.18	2.08	18.95	4.07	758.01	27.50	0.74
	60-10-Hyperband**	2131.60	2.12	19.36	4.16	972.96	31.19	0.74
XGBM	Default Settings	1855.62	2.17	14.88	3.20	620.32	24.91	0.68
LGBM	Default Settings	273.35	2.18	15.00	3.22	666.65	25.82	0.68
Naive	N/A	0.94	2.28	16.99	3.65	787.62	28.06	0.69
MLR	Default Settings	3.72	2.29	15.69	3.37	667.70	25.84	0.67
PyCaret	Default Settings	2853.39	2.40	16.02	3.44	738.44	27.17	0.64
Ludwig	Max. 100 Epochs	82.80	2.76	17.31	3.72	794.29	28.18	0.56
FLAML	Default Settings	3563.25	2.84	23.23	4.99	1,035.81	32.18	0.65
FEDOT	Best-quality	6306.42	2.92	22.72	4.88	926.00	30.43	0.62
Ludwig	Max. 150 Epochs	126.62	2.97	19.32	4.15	938.81	30.64	0.54
	Max. 200 Epochs	51.38	3.04	18.42	3.96	891.94	29.87	0.50
	Max. 50 Epochs	93.83	3.09	17.31	3.72	849.77	29.15	0.45
	Max. 10 Epochs	39.55	3.25	18.29	3.93	907.12	30.09	0.42
AutoGluon	High-quality	801.39	3.44	23.02	4.95	1135.94	33.70	0.49
	Best-quality	931.34	3.53	23.12	4.97	1134.47	33.68	0.46
ETNA	N/A	37.66	3.66	27.11	5.83	1299.93	36.05	0.51
FEDOT	Fast-train	5129.02	3.86	24.02	5.16	1126.09	33.56	0.38
	Stable	5232.86	3.91	23.86	5.13	1105.61	33.25	0.36
	TS	5110.50	3.94	23.99	5.15	1116.52	33.41	0.35
AutoTS	Fast-60***	347.01	4.08	21.50	4.62	1058.63	32.54	0.23
	Fast-120***	554.12	4.08	21.50	4.62	1058.63	32.54	0.23
	All-480***	4170.20	4.20	22.00	4.73	1126.11	33.56	0.20
	Default-480***	4176.00	4.20	22.00	4.73	1126.11	33.56	0.20
	Fast-parallel-480***	4138.52	4.20	22.00	4.73	1126.11	33.56	0.20
FEDOT	Auto	5951.96	5.07	25.91	5.57	1328.25	36.45	0.01
AutoGluon	Fast-training	192.71	N/A*	20.28	4.36	1038.02	32.22	N/A*
	Medium-quality	316.28	N/A*	20.28	4.36	1038.02	32.22	N/A*

* Some Naive forecasters output a constant value, which makes it impossible to calculate SRC.
** The AutoKeras '60-50-Greedy' preset refers to 60 trials, maximum 50 epochs, Greedy Optimiser.
*** The AutoTS preset Fast-parallel-480 refers to the "fast parallel" configuration with a 480-second GA generation time constraint.

The baseline models of XGBM, LGBM, Naive Forecasting and Linear Regression scored GM-MAE-SR scores of 2.17, 2.18, 2.28 and 3.72, respectively. As expected, the runtimes for the Naive and Linear models were short – 0.94 and 3.72 seconds, while XGBM and LGBM required approximately 31 and 4.5 minutes, respectively. The ranking of the baseline ML models is consistent with our previous study [6].

As the model hyperparameter optimisation process is identical to those used in [6], the higher MAE scores can be attributed to using a simpler feature set, i.e., only historical electricity prices and no oil prices, wind speeds, gas prices, etc.

As mentioned previously, this feature set was selected due to the high computational complexity of running AutoML algorithms.

AutoKeras was found to provide the most favourable, i.e., the lowest GM-MAE-SR of 1.91, although the training time took approximately 6.43 h. The best-performing result within the time limit achieved a GM-MAE-SR of 2.12 using the Hyperband optimiser. Unfortunately, AutoKeras is one of the more difficult libraries to configure as it requires users to specify epoch and trial limits with limited insight into the models' exploration process. The 13 worst GM-MAE-SR scores (omitted from Table III) were various configurations of AutoKeras.

Similarly, the performance of the AutoTS library is sensitive to the two GA parameters `max_generations` and `generation_timeout` with runtimes varying from ~5 minutes to ~1.15 h depending on the GA generation limit and maximum generation time parameters, suggesting that the maximum generation time parameter is not effectively applied internally. AutoTS does generate favourable GM-MAE-SR scores of 2.09 (Fast-parallel-480 and Default-480) and 2.12 (All-480) but only when significantly violating the time constraint. AutoTS is also memory-intensive, being the only library requiring more than 32 GB of RAM with the I-SEM dataset for the All preset. The performance of AutoTS was surprising given its efficacy in other applications [39]. To account for potential overfitting on the relatively small dataset as well as to reduce the overall runtime, a time limit parameter was introduced and enforced for the AutoTS functions `fit()` and `_run_template()`. The addition of a time limit parameter improved the mean GM-MAE-SR score of all presets from 5.18 to 4.84 and the mean duration from 19 281 to 3935 s. These improved results correspond to those described in Fig. 1 which illustrates the AutoTS improvements with respect to GM-MAE-SR and runtime duration.

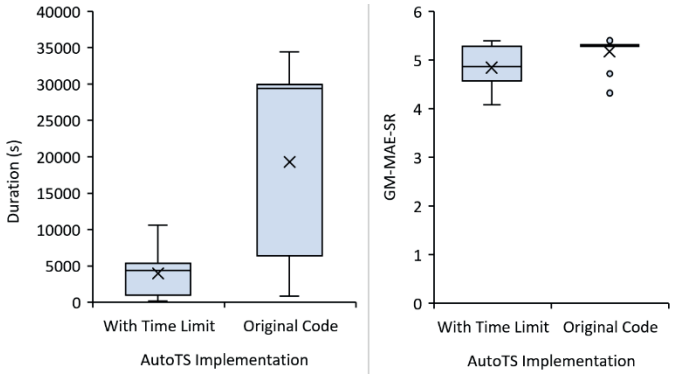


Fig. 1. AutoTS improvement with a time limit.

AutoGluon GM-MAE-SR scores were 3.44 and 3.53 for the High and Best quality presets, respectively, while the Medium and Low quality presets output constant values, thus failing to fit an appropriate model for scheduling even though their MAE scores are lower than the Best and High quality presets. This supports the finding that error metrics should be combined with ranking metrics to ensure suitable forecasting models for scheduling applications. Each of the presets reliably terminated in less than 20 min with the shortest being only ~3 min. ETNA's AutoML class *Auto* selected the `MovingAverageModel` (with lag = 48) class regardless of the values selected for the `tune_size` and `n_trials` parameters. As expected, larger numbers result in slower computation times but do not affect the result. Additionally, ETNA often fails to execute with internal library errors.

FEDOT achieved a GM-MAE-SR of 2.92 with the Best quality preset, which took 1.75 h. That the FEDOT library exceeded the time limit could imply that the library may produce more accurate forecasts under different experimental conditions, i.e., an increased time limit. To investigate this possibility, each preset of FEDOT was used to train forecasting models using a 2-hour time limit. Interestingly, the mean GM-MAE-SR increased from 4.07 to 4.69, which may have been the result of overfitting and a relatively small dataset. FEDOT outperformed AutoTS in this benchmark, which is consistent with their own work [30].

PyCaret and FLAML were executed with one configuration as their APIs required very little configuration to generate results from a variety of model types, making them user-friendly and accessible to early-stage programmers. Both libraries reliably terminated within the time limit and produced SRC values comparable with the baseline models (0.64 and 0.65, respectively), while their GM-MAE-SR values were marginally worse than the baseline models at 2.4 and 2.84. Curiously, the PyCaret library produced a worse result when using its `tune_model` function after initial training, which resulted in selecting a naive forecaster outputting a constant value.

Ludwig is more difficult to configure via the Python API due to sparse documentation and seemingly limited configuration options for its time series functionality. As a result, its GM-MAE-SR scores were worse than the baseline models ranging from 2.76 to 3.09. The low scores and very short runtimes – 51.38 to 126.62 s – suggest that early stopping was triggered early with little or no model exploration.

Overall, the extent to which the tested AutoML libraries conform to the expectations set by their respective documentations and API parameters varies considerably by library, particularly with respect to time limit parameters.

V. Conclusions

In this paper, an empirical benchmark of AutoML libraries with time series forecasting functionality was presented to evaluate the performance of eight Python-based software packages. These libraries were applied to the problem of day-ahead electricity price forecasting in the Irish electricity market I-SEM and examined using runtime and conventional error metrics such as MAE, MASE, etc. As electricity price forecasting is typically conducted with a view of optimising electricity-dependent operation profitability, the libraries are also evaluated using the geometric mean of MAE and SRC (GM-MAE-SR) to reflect the impact of correct price ranking in schedule generation.

While AutoKeras provides the most accurate forecasts, PyCaret, Ludwig, FLAML and FEDOT resulted in scores within 1 GM-MAE-SR of the best AutoKeras result. The PyCaret, FEDOT and FLAML frameworks are both appreciably convenient to use as they can generate forecasts of reasonable accuracy without extensive configuration. Conversely, AutoKeras and AutoTS require more ad-hoc experimentation to determine usable parameters as both libraries may be prohibitively slow depending on time constraints and the parameters selected. Ludwig is found to be time efficient but difficult to configure due to a lack of time series documentation both online and in existing source code.

ETNA produced the same result regardless of input parameters and often terminates with internal library errors across a range of software versions. Therefore, ETNA is excluded from Table IV, which provides a final summary of the library performances.

TABLE IV
AUTOML LIBRARY SUMMARY

Library	Rank		Ease-of-Use
	GM-MAE-SR	Duration	
AutoGluon	6	2	Accessible. Installation needs careful attention.
AutoKeras	1	7	Convenient to run, but difficult to configure due to long runtimes.
AutoTS	7	5	Convenient to run, but difficult to configure due to long runtimes.
FEDOT	5	6	Accessible.
FLAML	4	4	Highly accessible.
Ludwig	3	1	Documentation needs improvement.
PyCaret	2	3	Highly accessible.

VI. FUTURE WORK

AutoML class or function. There are other libraries with APIs providing tabular regression functionality that could be adapted to this problem by formatting a time series dataset using the methodology explained in this paper for the AutoKeras, AutoTS and FEDOT libraries. This could be used to expand the work to include libraries such as H2O and auto-sklearn but may require manual and extensive time series feature engineering to create a useful feature set.

The examined libraries were tested on data from the I-SEM day-ahead electricity price market, but benchmarks exist for more general model comparison and testing such as Libra [41] and the Monash Time Series Forecasting Archive [40]. In particular, larger datasets may result in more favourable results for libraries that make extensive use of neural network models, e.g., AutoKeras. Furthermore, the application of these AutoML libraries should be amenable to utilisation in specific use-case domain-agnostic designs. For example, financial datasets often contain a temporal component when tracking variables such as returns, Sharpe ratios, etc. The examined AutoML libraries may be of particular interest to researchers developing schedule-optimisation algorithms incorporating a forecasting element.

ACKNOWLEDGEMENT

Thanks are extended to MTU for their support via the Recurrent Funding Allocation Model (RFAM) program.

REFERENCES

[1] B. Donlon, "Quick guide to the integrated single electricity market," Tech. Rep., 2016. [Online]. Available: https://www.eirgridgroup. com/ uuid/1458bec2-f1e3-493c-92de-8dd2228bca1c/EirGrid-Group-I-SEM-Quick-Guide.pdf

[2] SEM-O, "Market operator performance." [Online]. Available: http://www.sem-o. com/publications/operator-performance/

[3] K. Kavanagh, M. Barrett, and M. Conlon, "Short-term electricity load forecasting for the integrated single electricity market (I-SEM)," in *2017 52nd International Universities Power Engineering Conference (UPEC)*, Heraklion, Greece, Aug. 2017, pp. 1–7. https://doi.org/10.1109/UPEC.2017.8231994

[4] S. Beltrán, A. Castro, I. Irizar, G. Naveran, and I. Yeregui, "Framework for collaborative intelligence in forecasting day-ahead electricity price," *Applied Energy*, vol. 306, Part A, Jan. 2022, Art. no. 118049. https://doi.org/10.1016/j.apenergy.2021.118049

[5] C. McHugh, S. Coleman, and D. Kerr, "Hourly electricity price forecasting with NARMAX," *Machine Learning with Applications*, vol. 9, Sep. 2022, Art. no. 100383. https://doi.org/10.1016/j.mlwa.2022.100383

[6] C. Lynch, C. O'Leary, P. G. K. Sundareshan, and Y. Akin, "Experimental analysis of GBM to expand the time horizon of Irish electricity price forecasts," *Energies*, vol. 14, no. 22, Nov. 2021, Art. no. 7587. https://doi.org/10.3390/en14227587

[7] C. O'Leary, C. Lynch, R. Bain, G. Smith, and D. Grimes, "A comparison of deep learning vs traditional machine learning for electricity price forecasting," in *2021 4th International Conference on Information and Computer Technologies (ICICT)*, HI, USA, Mar. 2021, pp. 6–12. https://doi.org/10.1109/ICICT52872.2021.00009

[8] D. Grimes, G. Ifrim, B. O'Sullivan, and H. Simonis, "Analyzing the impact of electricity price forecasting on energy cost-aware scheduling," *Sustainable Computing: Informatics and Systems*, vol. 4, no. 4, pp. 276–291, Dec. 2014. https://doi.org/10.1016/j.suscom.2014.08.009

[9] J. Lago, F. De Ridder, and B. De Schutter, "Forecasting spot electricity prices: Deep learning approaches and empirical comparison of traditional algorithms," *Applied Energy*, vol. 221, pp. 386–405, Jul. 2018. https://doi.org/10.1016/j.apenergy.2018.02.069

[10] A. R. Gollou and N. Ghadimi, "A new feature selection and hybrid forecast engine for day-ahead price forecasting of electricity markets," *Journal of Intelligent and Fuzzy Systems*, vol. 32, no. 6, pp. 4031–4045, May 2017. https://doi.org/10.3233/JIFS-152073

[11] L. Wang, Z. Zhang, and J. Chen, "Short-term electricity price forecasting with stacked denoising autoencoders," *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 2673–2681, Jul. 2017. https://doi.org/10.1109/TPWRS.2016.2628873

[12] S. K. Aggarwal, L. M. Saini, and A. Kumar, "Electricity price forecasting in deregulated markets: A review and evaluation," *International Journal of Electrical Power & Energy Systems*, vol. 31, no. 1, pp. 13–22, Jan. 2009. https://doi.org/10.1016/j.ijepes.2008.09.003

[13] U. Ugurlu, I. Oksuz, and O. Tas, "Electricity price forecasting using recurrent neural networks," *Energies*, vol. 11, no. 5, May 2018, Art. no. 1255. https://doi.org/10.3390/en11051255

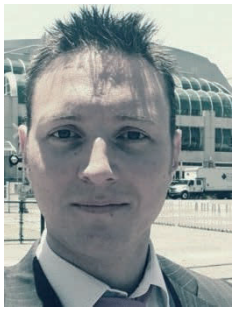
[14] F. Arci, J. Reilly, P. Li, K. Curran, and A. Belatreche, "Forecasting short-term wholesale prices on the Irish single electricity market," *Int. Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 6, pp. 4060–4078, Dec. 2018. https://doi.org/10.11591/ijece.v8i6.pp4060-4078

[15] F. Feijoo, W. Silva, and T. K. Das, "A computationally efficient electricity price forecasting model for real time energy markets," *Energy Conversion and Management*, vol. 113, pp. 27–35, Apr. 2016. https://doi.org/10.1016/j.enconman.2016.01.043

[16] G. Osório, J. Matias, and J. Catalão, "Electricity prices forecasting by a hybrid evolutionary-adaptive methodology," *Energy Conversion and Management*, vol. 80, pp. 363–373, Apr. 2014. https://doi.org/10.1016/j.enconman.2014.01.063

[17] P.-H. Kuo and C.-J. Huang, "An electricity price forecasting model by hybrid structured deep neural networks," *Sustainability*, vol. 10, no. 4, Apr. 2018, Art. no. 1280. https://doi.org/10.3390/su10041280
[18] C. O'Leary, "Capsule networks for electricity price forecasting," Master's thesis, Munster Technological University, 2020.
[19] A. Thessen, "Adoption of machine learning techniques in ecology and Earth science," *One Ecosystem*, vol. 1, Jun. 2016, Art. no. e8621. https://doi.org/10.3897/oneeco.1.e8621
[20] C. O'Leary, F. G. Toosi, and C. Lynch, "A review of AutoML software tools for time series forecasting and anomaly detection," in *Proceedings of the 15th International Conference on Agents and Artificial Intelligence (ICAART)*, Lisbon, Portugal, Oct. 2023, pp. 421–433. https://doi.org/10.5220/0011683000003393
[21] X. He, K. Zhao, and X. Chu, "AutoML: A survey of the state-of-the-art," *Knowledge-Based Systems*, vol. 212, Jan. 2021, Art. no. 106622. https://doi.org/10.1016/j.knosys.2020.106622
[22] I. Y. Javeri, M. Toutiaee, I. B. Arpinar, J. A. Miller, and T. W. Miller, "Improving neural networks for time-series forecasting using data augmentation and AutoML," in *2021 IEEE Seventh International Conference on Big Data Computing Service and Applications (BigDataService)*, Oxford, United Kingdom, Aug. 2021, pp. 1–8. https://doi.org/10.1109/BigDataService52369.2021.00006
[23] "Stack overflow developer survey 2023." [Online]. Available: https://survey.stackoverflow.co/2023/?utm source=social-share&utm medium=social&utm campaign= dev-survey-2023
[24] C. Lynch, J. Kehoe, R. Bain, F. Zhang, J. Flynn, C. O'Leary, G. Smith, R. Linger, K. Fitzgibbon, and F. Feijoo, "Application of a SVM-based model for day-ahead electricity price prediction for the single electricity market in Ireland," in *39th International Symposium on Forecasting (ISF)*, 2019.
[25] P. Li, F. Arci, J. Reilly, K. Curran, A. Belatreche, and Y. Shynkevich, "Predicting short-term wholesale prices on the Irish single electricity market with artificial neural networks," in *2017 28th Irish Signals and Systems Conference (ISSC)*, Killarney, Ireland, Jun. 2017, pp. 1–8. https://doi.org/10.1109/ISSC.2017.7983623
[26] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, and A. Smola, "AutoGluon-Tabular: robust and accurate AutoML for structured data," *arXiv:2003.06505*, Mar. 2020. https://doi.org/10.48550/arXiv.2003.06505
[27] H. Jin, Q. Song, and X. Hu, "Auto-Keras: An efficient neural architecture search system," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ser. KDD'19*, New York, NY, USA, Jul. 2019, pp. 1946–1956. https://doi.org/10.1145/3292500.3330648
[28] C. Catlin, "AutoTS," Aug. 2022, original-date: 201911-26T14:13:16Z. [Online]. Available: https://github.com/ winedarksea/AutoTS
[29] "tinkoff-ai/etna," Aug. 2022, original-date: 2021-0827T14:02:56Z. [Online]. Available: https://github.com/tinkoff-ai/etna
[30] N. O. Nikitin, P. Vychuzhanin, M. Sarafanov, I. S. Polonskaia, I. Revin, I. V. Barabanova, G. Maximov, A. V. Kalyuzhnaya, and A. Boukhanovsky, "Automated evolutionary approach for the design of composite machine learning pipelines," *Future Generation Computer Systems*, vol. 127, pp. 109–125, Feb. 2022. https://doi.org/10.1016/j.future.2021.08.022
[31] C. Wang, Q. Wu, M. Weimer, and E. E. Zhu, "FLAML: A fast and lightweight AutoML library," Apr. 2021. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2021/file/1ccc3bfa05cb37b917068778f3c4523a-Paper.pdf
[32] M. Ali, "PyCaret: An open source, low-code machine learning library in Python," 2020. [Online]. Available: https://pypi.org/project/pycaret/2.2.3/
[33] P. Molino, Y. Dudin, and S. S. Miryala, "Ludwig: a type-based declarative deep learning toolbox," *arXiv:1909.07930*, Sep. 2019. https://doi.org/10.48550/arXiv.1909.07930
[34] M. Feurer, K. Eggensperger, S. Falkner, M. Lindauer, and F. Hutter, "Auto-Sklearn 2.0: Hands-free AutoML via MetaLearning," *arXiv:2007.04074*, Sep. 2021. https://doi.org/10.48550/arXiv.2007.04074
[35] A. Vakhrushev, A. Ryzhkov, M. Savchenko, D. Simakov, R. Damdinov, and A. Tuzhilin, "LightAutoML: AutoML solution for a large financial services ecosystem," *arXiv:2109.01528*, Apr. 2022. https://doi.org/10.48550/arXiv.2109.01528
[36] "AxeldeRomblay / MLBox." [Online]. Available: https://github.com/AxeldeRomblay/MLBox
[37] "H2O.ai," 2022. [Online]. Available: https://github.com/h2oai/ h2o-3
[38] "alteryx/evalml," Aug. 2022, original-date: 2019-0717T21:36:30Z. [Online]. Available: https://github.com/alteryx/evalml
[39] S. Makridakis, C. Fry, F. Petropoulos, and E. Spiliotis, "The future of forecasting competitions: Design attributes and principles," *arXiv:2102.04879*, May 2021. https://doi.org/10.48550/arXiv.2102.04879
[40] R. Godahewa, C. Bergmeir, G. I. Webb, R. J. Hyndman, and P. Montero-Manso, "Monash time series forecasting archive," in *35th Conference on Neural Information Processing Systems*, 2021. [Online]. Available: https://openreview.net/pdf?id=I01l7rc0jcb
[41] A. Bauer, M. Zufle, S. Eismann, J. Grohmann, N. Herbst, and S. Kounev, "Libra: A benchmark for time series forecasting methods," in *Proceedings of the ACM/SPEC International Conference on Performance Engineering, ser. ICPE'21*, New York, NY, USA, Apr. 2021, pp. 189–200. https://doi.org/10.1145/3427921.3450241

Christian O'Leary is currently undertaking a PhD in automated machine learning at the Department of Computer Science at Munster Technological University (MTU) in Ireland. Christian completed his MSc in AI at MTU in 2020 on electricity price forecasting using machine learning and his BSc in Computer Science at University College Cork (UCC) in 2017 on search strategies for multi-agent systems. He joined the Nimbus Research Centre in MTU in 2017 and currently works as a Senior Researcher in the data science and AI team working on R&D projects in forecasting, anomaly detection, sentiment analysis, optimisation, and computer vision. He has previously worked as a Software Engineer at Tyco.
Christian O'Leary is currently a member of Lero, the SFI Research Centre for Software in Ireland.
Email: christian.oleary@mtu.ie
ORCID iD: https://orcid.org/0000-0002-9398-6950



Conor Lynch completed his PhD in model predictive control in 2016 at the Cork Institute of Technology (CIT) in Ireland in partnership with United Technologies Research Centre (UTRC) following two Level 8 honorary Bachelor of Engineering degrees in Structural Engineering and Sustainable Energy Technology in CIT in 2006 and 2011, respectively.
He joined the Nimbus Research Centre of Munster Technological University, Ireland in 2016 and is currently a Research Fellow managing the data science and AI team. During his PhD, he completed a term with the Applied Control Technology Consortium (ACTC) under the guidance of Prof Mike Grimble through the University of Strathclyde, Glasgow, Scotland. In that time, he studied System Modelling, Identification and Parameter Estimation Methods for the Predictive Control for Linear and Nonlinear Systems.
Conor Lynch is an academic member of Lero, the SFI Research Centre for Software in Ireland.
E-mail: conor.lynch@mtu.ie

Farshad Ghassemi Toosi earned his PhD in Computer Science from the University of Limerick in Ireland in 2017. His doctoral research focused on data visualisation, particularly in the domain of Graph Drawing, and the application of Genetic Algorithms in this field. Following the completion of his PhD, he served as a Post Doctorate for approximately two years, concentrating on Software Engineering with a specific emphasis on Source Code Manipulation and Feature Location.
Subsequently, he assumed the role of a full-time Lecturer at the Department of Computer Science of Munster Technological University in Cork starting from 2019. Farshad Ghassemi Toosi is an esteemed academic member of Lero, the SFI Research Centre for Software in Ireland.
E-mail: farshad.toosi@mtu.ie
ORCID iD: https://orcid.org/0000-0002-1105-4819

APPENDIX – AUTOML LIBRARY PRESETS

A. AutoGluon Presets

AutoGluon (v0.8.2) presets are described using their class names from `AutoGluon.timeseries.predictor.py` as follows:

- **`fast_training`**: ETS, Theta, Naive, Seasonal Naive and fast tree-based Recursive Tabular.
- **`medium_quality`**: "fast_training" models and Deep AR.
- **`high_quality`**: "medium_quality" models, Auto ETS, Auto ARIMA, tree-based model Direct Tabular, Temporal Fusion Transformer and Patch TST.
- **`best_quality`**: all previous models, more tabular models, multiple copies of Deep AR.

B. AutoKeras Models

Upon initialising the AutoKeras (v1.0.20) `TimeseriesForecaster` class, a `RegressionHead` block is constructed which is responsible for creating densely connected layers for regression with options for dropout and early stopping. The default regression loss metric is MSE. The `tuner` parameter accepts any one of `greedy`, `bayesian`, `hyperband` or `random`. These tuners initialise a `HyperModel` which defines the model search space using KerasTuner.

C. AutoTS Presets

For brevity, the forecasting methods implemented by AutoTS (v0.6.0) are listed here using their class names[9]:

- **`superfast`**: Constant Naive, Last Value Naive, Average Value Naive, GLS, Seasonal Naive and Seasonality Motif.
- **`fast`**: "superfast" and GLM, ETS, VAR, VECM, Window Regression, Datepart Regression, Univariate Motif, Multivariate Motif, NVAR, MAR, RRVAR, Kalman State Space, Metric Motif and Cassandra.
- **`fast_parallel`**: "fast" and FB Prophet, ARIMA, Unobserved Components, Theta, ARDL and ARCH11.
- **`default`**: GluonTS, Multivariate Regression, Sectional Motif, Univariate Regression and all previous models except MAR, RRVAR, Kalman State Space, Cassandra.
- **`all`**: Dynamic Factor, Dynamic Factor MQ, LATC, ML Ensemble, Motif Simulation, Neural Prophet, PyTorch Forecasting, Rolling Regression, TMF, and all previous models.

D. ETNA Presets

ETNA (v2.4.0) uses Optuna's `TPESampler` class by default to optimise pipelines. ETNA uses the following model classes for its `Auto` class: `NaiveModel`, `MovingAverageModel`, `SeasonalMovingAverageModel`, `HoltWintersModel`, `LinearPerSegmentModel`, `LinearMultiSegmentModel`, `ElasticPerSegmentModel`, `ElasticMultiSegmentModel`, `CatBoostMultiSegmentModel`, `CatBoostPerSegmentModel`, `ProphetModel`.

E. FEDOT Presets

The operations employed by FEDOT (v0.6.1) using the '`ts`' or time series preset for forecasting include AdaBoost regression, ARIMA, Autoregressive (AR), Decision Tree, Numerical Derivative Filter, Exponential Smoothing, Gaussian Filter, Generalized Linear Model (GLM), Fast ICA from scikit-learn, Lagged Transformation, Lasso regression, Linear regression, Naive average, Normalization, PCA, Polynomial, Repeat-Last-Value, Ridge regression, Scaling, Seasonal and Trend decomposition using Loess (STL) ARIMA, Sparse Lagged Transformation and Stochastic Gradient Descent with Warm Restarts.

The official documentation lists time series presets as follows:

- **`best_quality`***: All models that are available for this data type and task are used.*
- **`fast_train`***: Models that learn quickly. This includes pre-processing operations (data operations) that only reduce the dimensionality of the data but cannot increase it. For example, there are no polynomial features and one-hot encoding operations.*
- **`stable`***: The most reliable preset in which the most stable operations are included.*
- **`auto`***: Automatically determine which preset should be used.*
- **`gpu`***: Models that use GPU resources for computation.*
- **`ts`***: A special preset with models for time series forecasting tasks.*
- **`automl`***: A special preset with only AutoML libraries such as TPOT and H2O as operations.*

F. FLAML Models

The FLAML (v2.0.2) `AutoML` class with the task parameter set to `ts_forecast` uses the following models: XGBM, XGBM with Limited Depth, Random Forest, LGBM, Extra Trees, ARIMA, SARIMAX, Holt-Winters, CatBoost, Temporal Fusion Transformer, Facebook Prophet and Damped Local Trend.

G. Ludwig Models

Upon initialising the `LudwigModel` class, Ludwig (v0.6.4) initialises a PyTorch backend with GPU integration if available. The `train` method of `LudwigModel` pre-processes data, sets up callbacks and trains models. Batch sizes and learning rates are tuned automatically via `tune_batch_size` and `tune_learning_rate` respectively.

H. PyCaret Models

The list of models used by PyCaret (v3.0.0) in its `TSForecastingExperiment` class can be accessed via the `models()` function. The full list of models is: Naive Forecaster, Grand Means, Seasonal Naive, Polynomial Trend, ARIMA, Auto ARIMA, Exponential Smoothing, Croston, ETS, Theta, TBATS, BATS, Prophet, Linear Regression, Elastic Net, Ridge, Lasso, Least Angular Regressor, Lasso Least Angular Regressor, Bayesian Ridge, Huber, Passive Aggressive, Orthogonal Matching Pursuit, K-Neighbours, Decision Tree, Random Forest, Extra Trees, Gradient Boosting, AdaBoost, XGBM, LGBM, CatBoost Regressor.

[9] https://winedarksea.github.io/AutoTS/build/html/source/tutorial.html

[11] https://bashtage.github.io/arch/