Journal of Information and Data Management, 2024, 15:1, doi: 10.5753/jidm.2024.3369

Empirical Comparison of EEG Signal Classification Techniques through Genetic Programming-based AutoML: An Extended Study

Icaro M. Miranda [**Universidade de Brasília, Brazil** | ***icaro.miranda@aluno.unb.br***]
Claus de C. Aranha [**Tsukuba University, Japan** | ***caranha@cs.tsukuba.ac.jp***]
André C. P. L. F. de Carvalho [**Universidade de São Paulo, Brazil, Brazil** | ***andre@icmc.usp.br***]
Luís P. F. Garcia [**Universidade de Brasília, Brazil** | ***luis.garcia@unb.br***]

Department of Computer Science - CIC, Universidade de Brasília, Campus Darcy Ribeiro - Asa Norte, Brasília - DF, 70910-900, Brazil.

Received: 29 April 2023 • **Published:** 27 February 2024

Abstract Machine Learning (ML) applications using complex data often need multiple preprocessing techniques and predictive models to find a solution that meets their needs. In this context, Automated Machine Learning (AutoML) techniques help to provide automated data preparation and modeling and improve ML pipelines. AutoML can follow different strategies, among them Genetic Programming (GP). GP stands out for its ability to create pipelines of arbitrary format, with high interpretability and the ability to customize information from the data domain context. This paper presents a comparative study of two AutoML approaches optimized with GP for the time series classification problem and its characterization through four domain-based feature sets. We selected the Electroencephalogram (EEG) signals as a case of study due to their high complexity, spatial and temporal co-variance, and non-stationarity. Our data characterization shows that using only spectral or time-domain features is unsuitable for achieving high-performance pipelines. Our results reveal how AutoML can generate more accurate and interpretable solutions than the literature's complex or *ad hoc* models. The proposed approach facilitates the analysis of dimensional reduction through fitness convergence, tree depth, and generated features.

Keywords: AutoML, Classification, EEG, End-to-end Machine Learning, Genetic Programming, Sleep Spindles

1 Introduction

End-to-end Machine Learning (ML) experiments for real-world applications require a pipeline of steps, including understanding the application context, data exploration, data transformation and preprocessing, data modeling, model evaluation, and solution implementation [Azevedo and Santos, 2008]. Each step, implemented by a flow of techniques and algorithms, can be subjective and time-consuming. However, a significant part of this process can be automated using Automated Machine Learning (AutoML) [Zöller and Huber, 2021; Hutter *et al.*, 2019].

Designing a pipeline for machine learning can be challenging due to the many algorithms and data preprocessing techniques available and the complexity of many applications. Typically, a data scientist needs to manually design a sequence of procedures, which can be complex. However, this task can be simplified and made more efficient by using AutoML to automate the process [Hutter *et al.*, 2019].

Popular AutoML open-source frameworks, such as *Auto-sklearn* [Hutter *et al.*, 2019], use Bayesian optimization to find good pipelines in a search space by selecting and optimizing preprocessing operations and performing hyperparameter tuning on ML algorithms. Other frameworks use different approaches, e.g., H2O [LeDell and Poirier, 2020] employs random search and stacking of the best pipelines found, and TPOT (Tree-based Pipeline Optimization Tool) [Olson and Moore, 2019] uses Genetic Programming (GP) to build them.

GP's main advantages are easy parallelization, high customization, interpretability, control of computational resources, and insertion of application domain information [Zöller and Huber, 2021]. It allows the creation of expressions or computer programs from combinations of user-defined operators. Its flexibility makes solving problems in different contexts with different data types [Poli *et al.*, 2008], even time series analysis [Miranda *et al.*, 2019], more effortless.

A time series is a data type where observations show a time ordering - and they depend on each other based on this ordering [Bontempi *et al.*, 2012]. Regarding the use of traditional ML algorithms on these data, transforming the temporal features through descriptive measures is necessary to represent the problem most properly [Motamedi-Fakhr *et al.*, 2014].

In the context of bio-signals, e.g., Electroencephalogram (EEG) signals are particularly problematic time series due to their high complexity, spatial and temporal co-variance, and non-stationarity [Kevric and Subasi, 2017]. Consequently, the classification of EEG signals typically necessitates the involvement of a medical specialist. However, this process can be time-consuming, exhausting, and vulnerable to errors, biases, mood fluctuations, distractions, and variations in expertise [Bontempi *et al.*, 2012; Motamedi-Fakhr *et al.*, 2014]. Thus, this intricate process faces the potential for failure, as a consensus among experts may only sometimes be attainable [Weiner and Dang-Vu, 2016].

Problems involving EEG use multiple input channels and can generate up to 256 simultaneous signals from different parts of the scalp [Yamazaki *et al.*, 2012]. Although the literature guides which channels to use in various contexts, many still need to be analyzed. Also, wavelet transforms may decompose each signal into multiple levels depending on the preprocessing method [Hazarika *et al.*, 1997]. Furthermore, to utilize traditional machine learning algorithms on time series data, it is imperative to partition the data into samples and compute a representative set of features [Acharya *et al.*, 2019; Motamedi-Fakhr *et al.*, 2014; Ilyas *et al.*, 2015].

As a result, due to the many approaches available to solve the same problem, data scientists usually adopt a manual and empirical approach to construct their solution pipelines [Bontempi *et al.*, 2012; Geurts, 2001]. Thus, it is a tiring and subjective task, which reduces experimental reproducibility and is prone to overfitting in the generated models [Corradino and Bucolo, 2015]. Overfitting is also a common problem even in pipelines generated by AutoML solutions [Fabris and Freitas, 2019], making its identification and prevention critical factors for time series analysis problems.

This paper compares state-of-the-art AutoML tools with the GP-based AutoML for identifying events in EEG signals, which is considered a complex benchmark [Motamedi-Fakhr *et al.*, 2014]. Besides, we propose characterizing EEG data using four feature groups: Statistical, Spectral, Complexity, and Time Series bases. For such, we use the Sleep Spindles public database [Devuyst *et al.*, 2011] of EEG signals to analyze: the quality of pipelines found for classification metrics, the selected features, and the pipeline structure. In the experiments, we obtained, for several classification measures, ML pipelines whose predictive performance was higher than those found by the other tools.

The main contributions of this study, as demonstrated by the findings, can be outlined as follows: (1) The characterization of Sleep Spindle data reveals that only time-based or spectral features are inadequate for achieving high-performance pipelines; (2) a comparative analysis of performance, utilizing the F1-score, among various traditional machine learning (ML) algorithms applied to raw and PCA-transformed data, benchmarks from the literature, and the GP-based AutoML demonstrates that pipelines containing simple ML algorithms can surpass the performance of intricate, customized models; (3) the analysis of the convergence of the proposed GP-based AutoML algorithms to prevent overfitting, understand pipeline complexity, and reduce data dimensionality.

This paper extends our previous work, Genetic Programming-based AutoML for EEG Signal Classification - A Comparative Study [Miranda *et al.*, 2022], published initially at Knowledge Discovery, Mining and Learning (KDMiLe) 2022. The present version builds on the initial findings and further expands the discussion, incorporating new insights and developments. As a result, the extension of our paper showcases a more thorough, well-rounded, and insightful exploration of the techniques of AutoML and the new experiments we carried out.

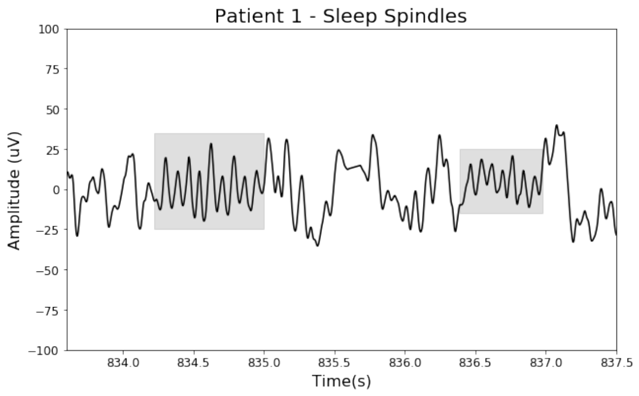

Figure 1. Two sleep spindles, marked in gray, identified in a 4-second window.

2 Background

This section briefly introduces the most fundamental GP concepts in the AutoML context. We also delve into AutoML algorithms, contrasting GP to other optimization methods and presenting their limitations. Finally, we discuss applications of EEG signals and mainly how it is possible to apply AutoML solutions in this domain.

2.1 Genetic Programming

GP is an evolutionary computation technique inspired by biological evolution for automatically creating computer programs that solve a given problem. GP iteratively transforms a set (called population) of computer programs through the heuristic search called evolution [Koza, 1994; Poli *et al.*, 2008].

At each iteration (called generation), GP stochastically modifies a set of possible solutions (called individuals) for the problem, generating new solutions by modifying and recombining the previous set. When appropriately parameterized, the algorithm enables the most recent individuals to outperform their predecessors in problem-solving. Furthermore, the stochastic nature of the technique allows for the evasion of local optima, to which deterministic methods typically converge [Eberhard *et al.*, 1999]. Therefore, GP is a technique that successfully develops new and unexpected ways to solve problems [Poli *et al.*, 2008]. Additionally, GP yields more interpretable outcomes, as the construction of individuals involves user-defined operations and constants. This characteristic facilitates fine-tuning through inspection.

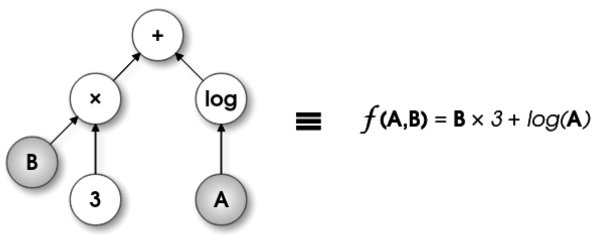
+
×
log
B
3
A
≡
$f(\mathbf{A},\mathbf{B}) = \mathbf{B} \times 3 + log(\mathbf{A})$

Figure 2. Example of a GP program

2.2 Representation

Individuals are hierarchical tree-like structures (Figure 2) composed of primitive functions and terminal values the user selects for the particular domain of the problem [Poli *et al.*, 2008]. The application context and the types of data involved affect the choice of these parameters. The set of primitive functions used can include, for example, arithmetic operations, mathematical functions, logical and conditional operations, and user-specified functions. The set of terminals can comprise numerical constants and specific entries for the problem [Koza, 1994]. For example, in Figure 2, the program represented has the variables A, B, and the constant value 3; and has the primitive functions of addition $+$, multiplication $\times$, and natural logarithm $\log$. Finally, we can rewrite it as the expression $B \times 3 + \log(A)$. Even though it is an entirely numerical example, GP is flexible for any input data as long as there are primitive functions, inputs, and compatible terminals.

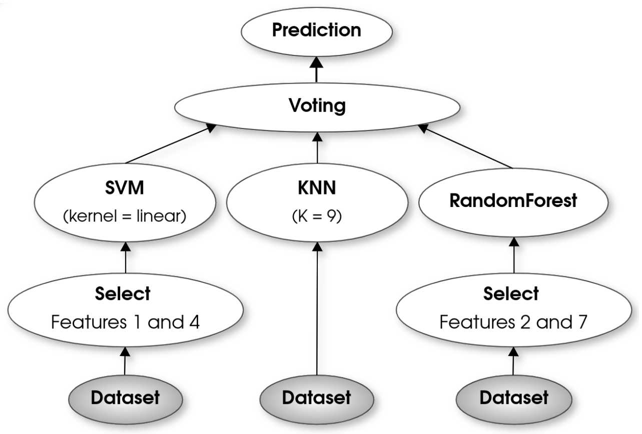

Figure 3. Example of pipeline represented by a GP tree

Naturally, in AutoML context, data processing functions and ML algorithms naturally constitute the components of the tree structure [Zöller and Huber, 2021]. Assuming the generation of the pipeline illustrated in Figure 3 over a dataset containing ten input columns and one label column, the algorithm executes a voting ensemble consisting of three machine learning models: Support Vector Machine (SVM), K-Nearest Neighbor (KNN), and Random Forest (RF). AutoML selects the hyperparameters, such as the kernel type for SVM and $K = 9$ for KNN. Moreover, the SVM and KNN models undergo training with the selected features.

2.3 Fitness Function and PG Initialization

With each generation of GP, a new population is created based on an attempt to reproduce the principle of natural selection, where the fittest individuals survive and are more likely to reproduce. This boils down to an assessment of the individual's quality, calculated through a fitness function, and its definition varies for each problem [Poli *et al.*, 2008], a crucial step in the preparation of the algorithm [Poli *et al.*, 2008]. In classification applications with AutoML, cost optimization functions or evaluation metrics such as AUC, F1-score, or balanced accuracy are usually used, as they reflect the quality of the obtained pipeline.

2.4 Bloat

Bloat is a phenomenon that happens during the execution of GP [Vanneschi *et al.*, 2010]. As generations progress, increasingly complex programs arise without enhancing their fitness value. This complexity often results from subtrees irrelevant to the problem or branches that could be simplified. In the context of AutoML, some pipelines may contain preprocessing operations and models that bring little or no performance improvement, for example, multiple occurrences of data normalization or overgrowth of an ensemble of classifiers. The event of bloat makes more processing necessary to evaluate the programs, decreasing the algorithm's performance and increasing the occurrence of overfitting [Vanneschi *et al.*, 2010; Gonçalves *et al.*, 2012].

2.5 Automated Machine Learning

The AutoML pipeline structure is modeled as a Directed Acyclic Graph (DAG). Each node represents a basic algorithm, and the edges represent the input data flow through the selected algorithms. In this way, the size of a particular pipeline, that is, its number of nodes (algorithms), can be used to indirectly measure a pipeline's complexity. Most DAGs that represent ML pipelines have the constraint that there is always a predictive algorithm at their last node.

The choice of algorithms that compose the pipeline depends mainly on the nature of the data and the problem goal. For example, in a classification application, it is necessary to have algorithms such as Naive Bayes (NB), KNN, Decision Trees (DT), and their hyperparameters in the search space to build the pipeline (preprocessing functions may be present in this set depending on the types of data involved). On the other hand, in the case of a numerical problem, for example, normalization, scaling, and transformation functions such as Principal Component Analysis (PCA) are used as candidates for pipeline elements [Gijsbers *et al.*, 2019].

Hence, this process is highly iterative and an ideal candidate for automation [Tuggener *et al.*, 2019]. Consequently, AutoML is an alternative to help specialists in Data Science by avoiding manual tasks, speeding up the process, and facilitating ML access for less specialized professionals [Xin *et al.*, 2021; Xanthopoulos *et al.*, 2020; Shang *et al.*, 2019]. Using AutoML, the data scientist is freed from these repetitive tasks and can focus on more creative tasks, adding more business value to the solution [Hutter *et al.*, 2019].

AutoML applications encompass techniques long explored in the automatic selection of algorithms and hyperparameters and feature extraction, selection, and construction [Zöller and Huber, 2021]. As computational resources become more accessible, along with affordable hardware, cloud solutions, and ML frameworks, a favorable environment emerges for the development and implementation of AutoML techniques [Wang *et al.*, 2009]. A factor that also helps in this process is expanding access to several open databases on platforms such as *Kaggle*[1] and UCI[2] [Dua and Graff, 2017]. Therefore, obtaining and using larger datasets

[1]`https://www.kaggle.com/`
[2]`http://archive.ics.uci.edu/ml`

is increasingly common in the number of samples and features.

However, difficulties arise when the data has many features/dimensions since a large sample size is needed to represent high-dimensional sample distributions. Bellman [Bellman *et al.*, 1957] defined this behavior as the "Curse of Dimensionality". In contrast, as Hughes [Hughes, 1968] shows, using numerous features can reduce classification and regression models' assertiveness and generalization capacity even when processing them requires computational power. So, Bellman and Hughes's concepts help us understand the difficulty of generalizing complex problems. Furthermore, the performance of several ML algorithms is sensitive to design decisions, which can be a problematic factor for less experienced users. Naturally, these difficulties get worse with the construction of AutoML pipelines.

Several widely used open-source AutoML tools are available, and each implements this concept through different strategies. Auto-sklearn, for example, uses Bayesian optimization to find the best algorithms and their hyperparameters in a pipeline involving preprocessing and modeling functions. These pipelines have a fixed structure formed by data cleaning, preprocessing, and modeling. In parallel, the H2O AutoML framework employs a random grid search to discover the hyperparameters of a model without incorporating preprocessing. In the end, it combines the best models using an ensemble.

Finally, the TPOT uses GP to build high-performance pipelines with variable structures for classification or regression problems, providing less overfitting than other tools [Fabris and Freitas, 2019]. In this context, the flexible tree structure properties, ease of customization of the nodes, and the very process of evolution of pipelines make GP quite attractive.

The three mentioned frameworks have different strategies for constructing their pipelines when optimizing their solutions. For example, while H2O AutoML generates pipelines with only one predictive layer, Auto-sklearn builds more complex DAGs, despite its fixed structure [Zöller and Huber, 2021; LeDell and Poirier, 2020; Hutter *et al.*, 2014]. In contrast, TPOT works with DAGs of arbitrary format.

Limiting the universe of pipeline structures reduces the problem's search space. In the context of H2O, which employs a computationally demanding optimization strategy, generating simpler DAGs ensures the evaluation of more solutions [LeDell and Poirier, 2020]. Despite the more complex search space, TPOT does not seek an exhaustive evaluation of all hyperparameters combinations, as it uses GP to optimize the algorithms.

AutoML solutions can ensure horizontal scalability and greater control of computing resources [Ferreira *et al.*, 2021; Olson and Moore, 2019]. On the other hand, as it involves algorithms of different natures and the evaluation of multiple pipelines simultaneously, the execution of AutoML routines often causes system failures due to intensive computational loads [Xin *et al.*, 2021]

Also, most commercial frameworks like Google Cloud AutoML [Bisong, 2019b,a; Drozdal *et al.*, 2020], for example, lack customization settings. As a result, the generated solutions do not have transparency during execution and interpretability, often treated as a black box [Xin *et al.*, 2021; Bosch *et al.*, 2021; Arzani *et al.*, 2021].

3 Related Work

Sleep Spindles (Figure 1), visible waveforms in the EEG signal during the polysomnography examination, are essential in sleep staging and identifying pathologies [Devuyst *et al.*, 2011; Clemens *et al.*, 2005; Niedermeyer and Ribeiro, 2000; Iranmanesh and Rodriguez-Villegas, 2017]. Despite a wide variety of recommended features for describing EEG signals [Motamedi-Fakhr *et al.*, 2014], characterization studies of Sleep Spindles data mainly focus on instantaneous statistics of the signal over the spectral and time domains [Ahmed *et al.*, 2009; Gomez-Pilar *et al.*, 2021; O'Reilly and Nielsen, 2014; Purcell *et al.*, 2017]. In a more general perspective, Lubba et al. [Lubba *et al.*, 2019] present a characterization of multiple time series problems - including some EEG datasets - to find a canonical set of features.

In order to use this kind of application, it is necessary to divide the signal into segments and calculate features for each created sample. These are varied, but it is possible to define four major groups of features: statistical, spectral, time-frequency, and non-linear [Motamedi-Fakhr *et al.*, 2014]. Despite this, finding a descriptive set to separate the samples into classes can be challenging. For example, in a problem with a five-channel EEG, where we decompose the signal into five wavelet transform levels and calculate ten features per sample, we obtain a database with 250 features. In this context, AutoML can help obtain more generalist pipelines that automatically discover the most relevant features. Our experiments show that we can find less complex and more performant predictive pipelines using more robust characterization measures than literature baselines using fewer data dimensions.

Despite the many features needed to characterize Sleep Spindles and the variety of characterization measures, many papers focus on a few *ad hoc* metrics, mainly instantaneous statistics on the signal in the time and spectral domains. Our experiments show that we can find less complex and more performant predictive pipelines using more robust characterization measures than literature baselines using fewer data dimensions.

The literature suggests several approaches for the Sleep Spindles data. Tsanas et al. [Tsanas and Clifford, 2015] and Zhuang et al. [Zhuang *et al.*, 2016] proposed Continuous Wavelet Transform (CWT) based approaches and the estimation of the probability of spindles occurrences. Lachner-Piza et al. [Lachner-Piza *et al.*, 2018] proposed a Support Vector Classifier (SVC) approach with an *ad hoc* supervised feature selection method based on correlations for determining the importance of each one. Finally, in previous work [Miranda *et al.*, 2019], we presented an automatic feature selection and construction GP-based algorithm to improve simple ML classifiers' performance.

Our previous method uses GP to evolve feature trees to improve the performance of the desired classifier. It builds a new dataset from the original data composed of combinations of the original features. Our GP combines them

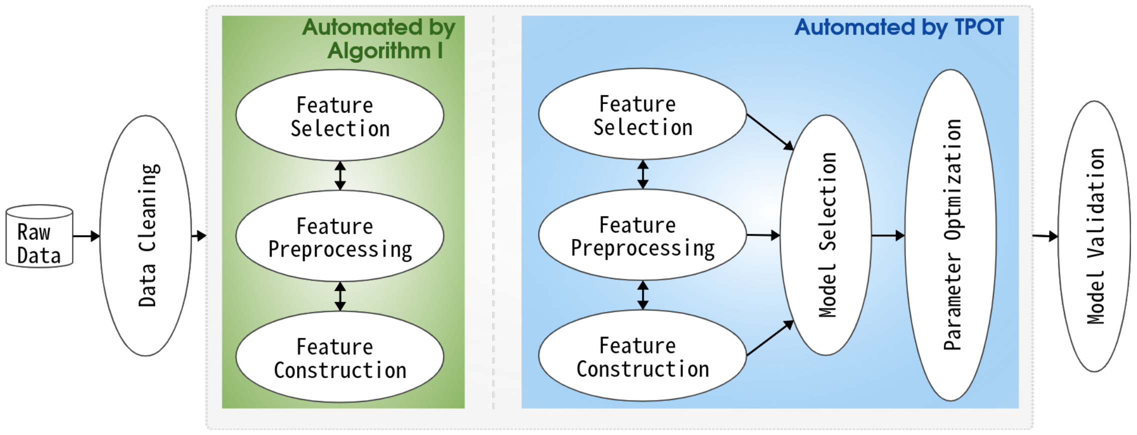

Figure 4. AutoML tasks automated by the two analyzed frameworks

through mathematical operators, allowing the addition of non-linearity through truncated mathematical operators, for example. From an AutoML perspective, this method automates attribute selection, creation, and preprocessing. We used the EEG data of Sleep Spindles and K-complexes as an object of study, showing that GP can empower simple classifiers by providing more sophisticated attribute engineering while reducing dimensionality and improving classification results [Miranda *et al.*, 2019].

The literature contains many GP-based AutoML applications that optimize different parts of the Data Science flow [Miranda *et al.*, 2019; Suchopárová and Neruda, 2020; de Sá *et al.*, 2018; Tran *et al.*, 2016; Guo *et al.*, 2011], such as algorithm selection, feature engineering, and hyperparameter optimization. As a complete AutoML solution, the framework TPOT [Olson *et al.*, 2016] is one of the most famous GP-based techniques.

TPOT considers preprocessing and classification techniques as GP operators and the dataset as a terminal to combine them in a pipeline. In addition to the restriction that every pipeline must have a classifier as the final operator, it is possible to construct them with an arbitrary format. In this way, GP trees provide a flexible representation for ML pipelines [Olson and Moore, 2019].

Despite its advantages, running TPOT on large datasets can be time-consuming [Zöller and Huber, 2021]. Due to the algorithm considers multiple ML algorithms in a pipeline with several preprocessing steps, the hyperparameters for all models, and various ways to group or stack algorithms in the pipeline. This problem is not unique to AutoML applications with GP, but the occurrence of bloat in pipelines can exacerbate it [Cerrada *et al.*, 2022].

Still, we can find applications of AutoML with promising results on EEG data. For example, Mei et al. [Mei *et al.*, 2017] used the framework TPOT to identify phenomena in EEG sleep, while Yang et al. [Yang *et al.*, 2021] analyzed the EEG of patients who suffered cardiac arrest.

4 Methodology

In order to study the application of GP-based AutoML techniques on the EEG classification problem, we performed a series of experiments comparing the performance of the algorithm TPOT with our previous work GP approach for feature engineering [Miranda *et al.*, 2019]. From now on, to facilitate our comparisons, we will name our previous work, Algorithm I. Although both are AutoML techniques and use GP as an optimization strategy, each automates different parts of pipeline creation. While TPOT proposes multi-step automation, our previous work focuses on constructing and selecting attributes. Figure 4 shows which ML steps Algorithm I and TPOT automate.

To carry out our study, we used the Sleep Spindle database of the DREAMS project [Devuyst *et al.*, 2011]. It consists of signals with expert notes on sleep phenomena or disorders. The dataset has 30-minute 3-channel EEG samples from eight patients, independently annotated by two sleep specialists. It is essential to highlight that all records in this database are from patients with sleep disorders, making identifying the phenomenon more challenging [Devuyst *et al.*, 2006].

We segmented the EEG signals into 2-second samples based on the maximum duration of 1.67s of the events identified in the data [Patti *et al.*, 2014; Al-Salman *et al.*, 2019]. In the sequence, we extracted samples from the original signals using the sliding window method with a 75% overlap [Parekh *et al.*, 2015; Patti *et al.*, 2015]. Next, we decomposed each sample into five levels of Discrete Wavelet Transform (DWT) db5 [Unser and Aldroubi, 1996; Amin *et al.*, 2015; Hazarika *et al.*, 1997]. Lastly, we calculated different sample feature groups to generate four new databases.

Table 1, describes the features of the four groups. Applications of EEG sleep signals already utilize the first three groups: statistical, spectral, and complexity [Motamedi-Fakhr *et al.*, 2014]. However, due to the wide variety of features available in this domain, understanding which groups aggregate the most relevant information to the problem can generate leaner and more generalist models. Lastly, the group catch22 comprises 22 features considered canonical for classification problems and time series clustering, obtained through applying multiple feature engineering techniques on 93 different classification problems [Lubba *et al.*, 2019]. The study of this group on EEG data allows for comparing the performance of AutoML techniques when evaluating features more or less specific to the application domain.

The spindle identification problem is unbalanced. Less than 2.5% of the total signal from the data are spindle events.

Table 1. Feature groups extracted from EEG data

Group	Features	Signal Domain	# Features
Statistic	{5, 25, 75, 95} percentile, mean, median, variance, entropy, RMS, STD, kurtosis, {zero, mean} crossings	Time	195
Spectral	{5, 25, 75, 95} percentile, mean, median, variance, entropy, RMS, STD, kurtosis, {zero, mean} crossings	Frequency	195
Complexity	{Shannon, FFT, SVD, Fisher} entropy, {Higushi, Pretosian} fractal dimension	Time/Frequency	90
Catch22	Constructed non-trivial time-frequency features	Time/Frequency	330

Table 2. AutoML techniques hyperparameters

	Previous Work	TPOT
Operator set	$\times, +, -, \div$, $\log(\lvert x\rvert)$, $\sqrt{\lvert x\rvert}$	Binarizer, Normalizer, VarianceThreshold, PCA, {MaxAbs, Robust, Standard}Scaler, Select{Fwe, Percentile}, RBFSampler, ZeroCount, FeatureAgglomeration
Classifier	Gaussian NB, DT RF, SV, KNN	Gaussian NB, Bernoulli NB, Multinomial NB, DT, KNN, Logistic regression (LR)
Hiper-parameters	Default (*scikit-learn*)	GP optimization
Terminal set	$-0.5, +0.5, -1, +1$	-

To mitigate this problem and maintain the unbalanced nature of the application domain, we reduced the majority class randomly until reaching the proportion of 70% of the samples in all four constructed datasets only for the training and validation sets.

For each group of features, we applied a cross-validation per patient. That is, patients selected for training did not have their samples tested. Thus, each group was performed eight times, with seven patients for training and validation and the last for testing.

Because of the techniques based on GP, we kept the same hyperparameters whenever possible: 100 for population size, 100 generations, F1-score for fitness function, 70% for crossover rate, 30% for mutation rate, and a time limit of 100 minutes. We show the divergent hyperparameters in Table 2. Also we maintained the same crossover and mutation rates as our previous study for the GP techniques. Algorithm I and TPOT algorithms operate with high mutation rates, given the high amplitude of the search space and the small changes caused by the kind of mutation operator in these approaches.

In order to observe the advantages of applying AutoML to the problem, we built a comparison basis through training and evaluation of five simple ML classifiers: DT, KNN, Gaussian NB, RF, and SVC. We evaluated these algorithms on the four groups of features without performing any additional preprocessing. In addition, we collected convergence metrics on training, validation, and test data about the pipeline's classification performance (confusion matrix) and complexity (pipeline size and the number of features) during the training. After that, we applied PCA to the groups of features to observe how the classifiers deal with the transformed features and dimensionality reduction keeping 99.5% of the explained variance proportion. Next, we trained the five classifiers using the PCA-transformed dataset and collected the same metrics from the previous step. Finally, we trained and evaluated Algorithm I and TPOT over the raw groups of features.

Using the performance and convergence metrics of the models, we compared the quality of the pipelines with multiple metrics and models from the literature trained on the same data. In addition, we presented the characterization of the Sleep Spindles on the feature groups, observing the performance and complexity of the pipelines obtained on each of them. We also discussed the increasing complexity of pipelines as the GP population converges. Finally, we discussed about the dimensionality reduction provided by each algorithm, observing the feature selection and construction operations. The following section presents all these results and analyses.

5 Experimental Results

In the literature, we can easily find references that use the DREAMS project databases to study the classification of EEG signals. We compared some references (Table 3) with multiple rating metrics (Sensitivity, Recall, Accuracy, and F1-score) with the obtained results. In Table 3, we present four algorithms discussed in Section 3 and their performance on Sleep Spindle data compared to the two AutoML approaches used in the experiments. Even though the results are individually competitive with the literature on Recall and Specificity, Algorithm I and TPOT presented a higher av-

Table 3. Comparison of the results obtained with models from the literature

Reference	Recall	Specificity	Precision	F_1score
[Tsanas and Clifford, 2015]	0.76	0.92	0.33	0.46
[Zhuang *et al.*, 2016]	0.51	0.99	0.70	0.59
[Lachner-Piza *et al.*, 2018]	0.65	0.98	0.38	0.48
[Miranda *et al.*, 2019]	0.75	0.98	0.35	0.48
Experiment (Algorithm I)	0.69 ± 0.09	0.92 ± 0.06	0.68 ± 0.19	0.65 ± 0.11
Experiment (TPOT)	0.67 ± 0.07	0.95 ± 0.03	0.73 ± 0.18	0.68 ± 0.09

erage F1-score than the other analyzed references, showing a better trade-off in rejecting false positives and false negatives.

To observe the performance of AutoML in contrast to the other classifiers, we compared the F1-score of Algorithm I and TPOT on raw data against ML classifiers on raw and PCA-transformed data. In Figure 5, we present the performance of each classifier or AutoML technique.

Figure 5 is composed of three grouped bar plots. Each classifier or AutoML technique was evaluated on the four groups of features (represented in different colors) with the metric F1-score, represented by the vertical bars and their respective confidence intervals (calculated over the cross-validation results).

We trained Algorithm I (which needs a target classifier the user selects) on the same classifiers used on raw data for a more direct comparison. Our results show that the performance of the algorithms drops when reducing the dimensionality with PCA, which shows the difficulty in eliminating features while maintaining the ability to discriminate the classes of the problem. Still, in Figure 5, we can observe that when applying AutoML techniques, there is an increase in the F1-score for all previously tested algorithms, including the case of SVC, which did not converge in the previous step. The TPOT algorithm, which builds more general pipelines with less focus on feature engineering, obtained competitive results. In addition, it stands out in the use of the features of the Spectral group.

When analyzing the evolution of the fitness function in Figure 7, we can observe a sharp growth in the initial generations followed by a slight increase until the end of the execution. Due to the established limit of 100 minutes, TPOT interrupted its execution before 100 generations. Both techniques converge to low values of the F1-score in the Spectral and Statistical groups, showing that the algorithms have difficulty adjusting pipelines capable of differentiating the problem classes. On the other hand, the Complexity and Catch22 groups proved to be more suitable for characterizing the problem.

To observe the increase in pipeline complexity, we analyzed the average number of operators in the population at each generation for TPOT and its equivalent for Algorithm I, represented by the average depth of individuals. Figure 6 shows the evolution of the average complexity of the pipelines for each feature group over the generations.

With the increase in fitness, we can also observe an increase in the complexity of the pipelines in Figure 6. To obtain solutions with higher performance, GP increments the pipelines with more features in the case of Algorithm I and

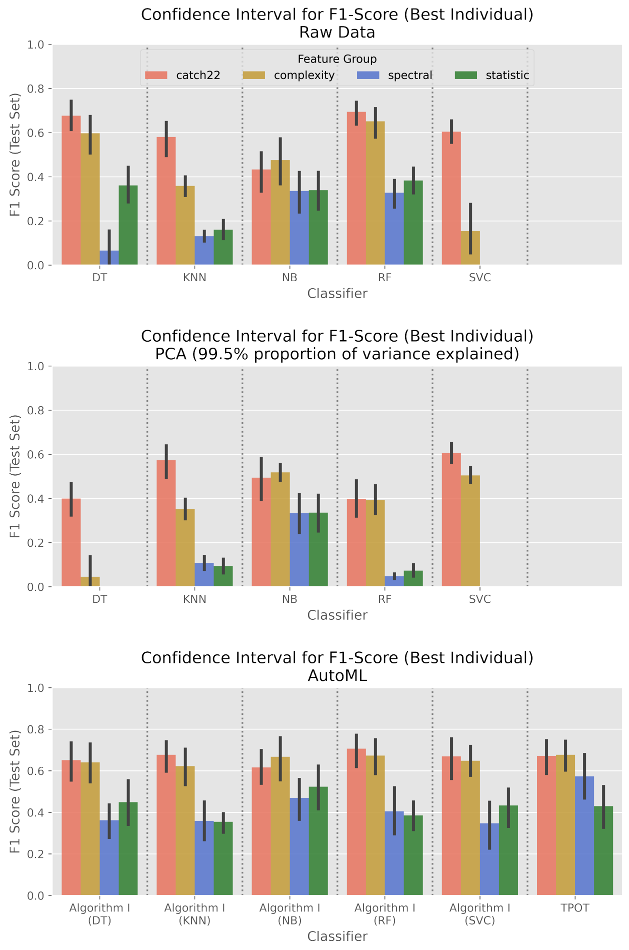

Figure 5. Evaluation of traditional ML models on raw data (top), traditional ML models on PCA-transformed data (middle) and F1-score obtained from AutoML pipelines on feature groups through cross-validation (bottom). Each classifier or AutoML technique was evaluated on the four groups of features (represented in different colors) with the metric F1-score, represented by the vertical bars and their respective confidence intervals (calculated over the cross-validation results).

with more operators in the case of TPOT. However, the increase in complexity does not always bring relevant improvements in performance. By observing the best trade-off between pipeline complexity and individual fitness, we can define a criterion for early stopping the algorithm.

In the case of TPOT, the number of operators in the constructed pipelines grows linearly, while fitness does not increase to the same extent. The situation is similar for Algorithm I. This undue increase in model learning capacity can lead to overfitting.

Although Algorithm I and TPOT are AutoML techniques, each automates different ML tasks. For example, when analyzing the content of the constructed pipelines, Algo-

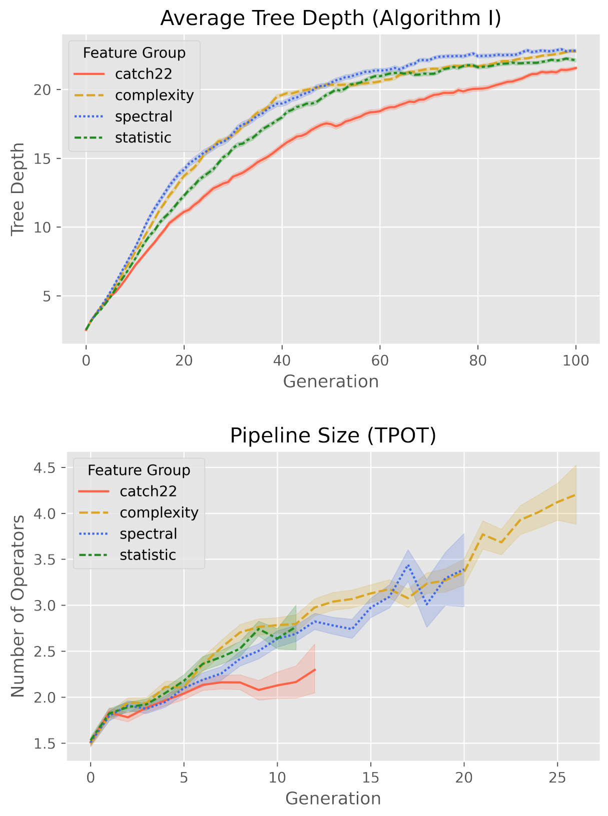

Figure 6. Increase in the complexity of individuals (pipelines) over generations.

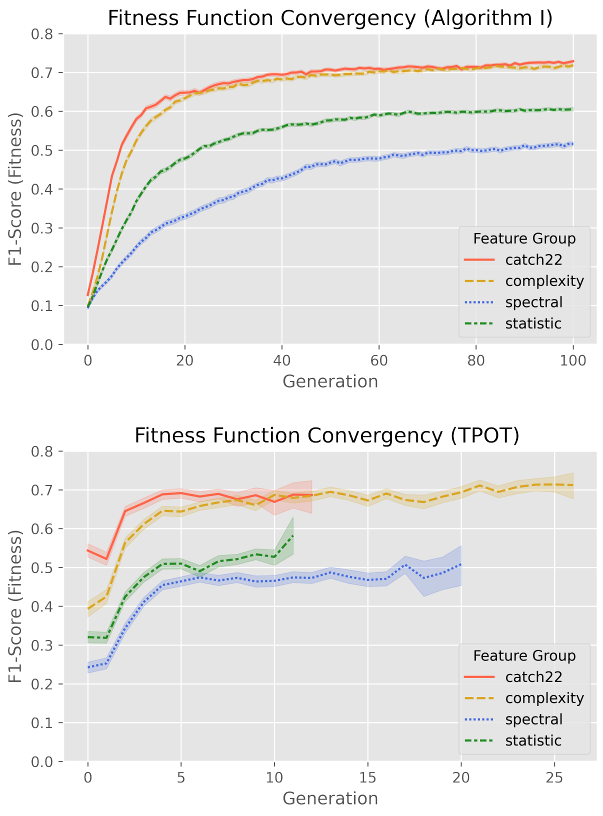

Figure 7. Convergence of the fitness function (population) on the validation data

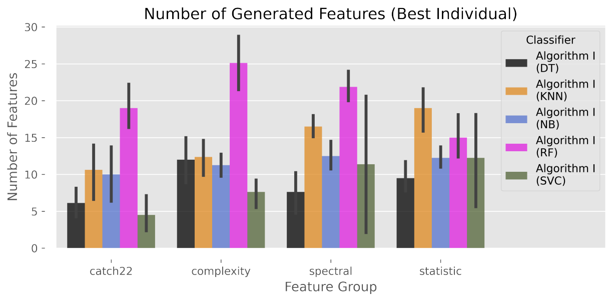

Figure 8. Number of features per group obtained with Algorithm I.

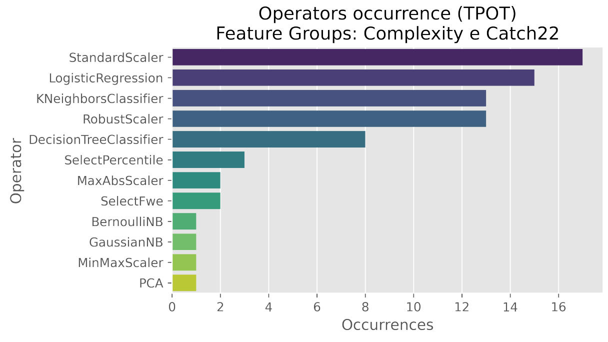

Figure 9. Operators found in the pipelines generated by TPOT.

rithm I provides a high dimensionality reduction for all feature groups regardless of the classifier used, as shown in Figure 8. In this Figure, the grouped bar chart shows the average number of attributes generated for each group of attributes through Algorithm I. As this technique uses a target classifier, the number of constructed attributes may vary according to the choice. Despite this variation, Algorithm I can bring up to a 97% reduction in dimensionality compared to the number of attributes of the raw data presented in Table 1.

For TPOT, it is not possible to measure dimensionality reduction so explicitly. However, as an indirect measure, in Figure 9, we present the number of occurrences per algorithm in the pipelines. So, it is possible to observe the TPOT's most frequent choices for the classification problem. Furthermore, when analyzing the complexity groups and catch22, TPOT selects a few feature reduction operators, such as PCA, *SelectPercentile*, and *SelectFwe* - predominantly, so its solutions work with all dimensions.

6 Conclusions

Time series analysis applications, particularly EEG signals, are challenging ML applications. For the analysis of EEG signals, the signals need to be segmented. Signal segmentation requires a pipeline of several stages when various transformations, cleaning functions, and, mainly, feature extraction techniques. Optimizing the pipeline in this variety of tasks generates a complex search space to be explored, especially for data scientists who do not have in-depth knowledge of the application domain.

Therefore, the application of AutoML techniques helps in this process by providing the data preparation, modeling, and evaluation of ML pipelines in an automated way. In addition to reducing the data scientist's participation in repetitive tasks, we can obtain more general solutions than a manual and empirical methodology, given the optimization strategies of AutoML.

In this comparative study, we investigated through systematic experiments the characterization of Sleep Spindle data, the performance comparison on the F1-score between literature baselines and GP-based AutoML, and the analysis of the convergence of the algorithms to avoid overfitting, understand pipeline complexity, and reduce data dimensionality. The results showed that the use of AutoML on the sleep EEG signals classification problem can generate more accurate solutions than complex models in the literature and also quantifies the importance of each group of features used. Furthermore, the analyzed techniques Algorithm I and TPOT can build more accurate classifiers than models from the literature through simple operators and classifiers. In addition, we collected evidence that using complexity measures and constructed features can better characterize of sleep spindles, improving the performance of the classifiers used in the raw data.

Both techniques showed a disproportionate increase in the complexity of the solutions over their performance, consequently causing overfitting problems and an increase in the time needed to perform a prediction.

There is still room to improve strategies from GP to AutoML. TPOT could not evolve its pipelines by the desired number of generations because it reached the time limit. In addition, the algorithm was interrupted in some executions due to inadequate management of computational resources. Despite superior results to models in the literature, their solutions can perform only simple transformations on the data. In parallel, Algorithm I performs a complex feature engineering and dimensionality reduction. However, its pipelines use only one classifier and cannot select preprocessing functions.

In future work, we want to extend this study to other AutoML approaches and more EEG contexts. So, in the following steps, we will also study the behavior of Bayesian optimization strategies and grid search on multiple EEG signal classification problems, applying the same experimental methodology. Therefore, we can classify each strategy based on its performance, the complexity of the obtained pipeline, interpretability, and the likelihood of overfitting. By evaluating these aspects, it will be possible to show the advantages of GP optimization over other strategies.

Authors' Contributions

IM carried out the experiments and, as the main contributor, wrote the manuscript with support from LG, CA, and AC. IM performed the experiments with the guidance of LG and CA. All authors read and approved the final manuscript. LG e CA supervised the project

Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

References

Acharya, U. R., Hagiwara, Y., Deshpande, S. N., Suren, S., Koh, J. E. W., Oh, S. L., Arunkumar, N., Ciaccio, E. J., and Lim, C. M. (2019). Characterization of focal eeg signals: a review. *Future Generation Computer Systems*, 91:290–299.

Ahmed, B., Redissi, A., and Tafreshi, R. (2009). A characterization of sleep spindles in eeg. In *World Congress on Medical Physics and Biomedical Engineering, September 7 - 12, 2009, Munich, Germany*. Springer Berlin Heidelberg.

Al-Salman, W., Li, Y., and Wen, P. (2019). Detecting sleep spindles in eegs using wavelet fourier analysis and statistical features. *Biomedical Signal Processing and Control*, 48:80–92.

Amin, H. U., Malik, A. S., Ahmad, R. F., Badruddin, N., Kamel, N., Hussain, M., and Chooi, W.-T. (2015). Feature extraction and classification for eeg signals using wavelet transform and machine learning techniques. *Australasian physical & engineering sciences in medicine*, 38(1):139–149.

Arzani, B., Hsieh, K., and Chen, H. (2021). Interpretable feedback for automl and a proposal for domain-customized automl for networking. In *Proceedings of the Twentieth ACM Workshop on Hot Topics in Networks*, pages 53–60.

Azevedo, A. I. R. L. and Santos, M. F. (2008). Kdd, semma and crisp-dm: a parallel overview. *IADS-DM*.

Bellman, R. E. *et al.* (1957). Dynamic programming, ser. *Cambridge Studies in Speech Science and Communication. Princeton University Press, Princeton*.

Bisong, E. (2019a). Google automl: cloud vision. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 581–598. Springer.

Bisong, E. (2019b). An overview of google cloud platform services. *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 7–10.

Bontempi, G., Taieb, S. B., and Le Borgne, Y.-A. (2012). Machine learning strategies for time series forecasting. In *European business intelligence summer school*, pages 62–77. Springer.

Bosch, N. *et al.* (2021). Automl feature engineering for student modeling yields high accuracy, but limited interpretability. *Journal of Educational Data Mining*, 13(2):55–79.

Cerrada, M., Trujillo, L., Hernández, D. E., Correa Zevallos, H. A., Macancela, J. C., Cabrera, D., and Vinicio Sánchez, R. (2022). Automl for feature selection and model tuning applied to fault severity diagnosis in spur gearboxes. *Mathematical and Computational Applications*, 27(1):6.

Clemens, Z., Fabo, D., and Halasz, P. (2005). Overnight verbal memory retention correlates with the number of sleep spindles. *Neuroscience*, 132(2):529–535.

Corradino, C. and Bucolo, M. (2015). Automatic preprocessing of eeg signals in long time scale. In *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 4110–4113. IEEE.

de Sá, A. G., Freitas, A. A., and Pappa, G. L. (2018). Automated selection and configuration of multi-label classification algorithms with grammar-based genetic programming. In *International Conference on Parallel Problem*

Solving from Nature, pages 308–320. Springer.

Devuyst, S., Dutoit, T., Didier, J.-F., Meers, F., Stanus, E., Stenuit, P., and Kerkhofs, M. (2006). Automatic sleep spindle detection in patients with sleep disorders. In *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 3883–3886. IEEE.

Devuyst, S., Dutoit, T., Stenuit, P., and Kerkhofs, M. (2011). Automatic sleep spindles detection—overview and development of a standard proposal assessment method. In *2011 Annual international conference of the IEEE engineering in medicine and biology society*, pages 1713–1716. IEEE.

Drozdal, J., Weisz, J., Wang, D., Dass, G., Yao, B., Zhao, C., Muller, M., Ju, L., and Su, H. (2020). Trust in automl: exploring information needs for establishing trust in automated machine learning systems. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*, pages 297–307.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Eberhard, P., Schiehlen, W., and Bestle, D. (1999). Some advantages of stochastic methods in multicriteria optimization of multibody systems. *Archive of Applied Mechanics*, 69(8):543–554.

Fabris, F. and Freitas, A. A. (2019). Analysing the overfit of the auto-sklearn automated machine learning tool. In *International Conference on Machine Learning, Optimization, and Data Science*, pages 508–520. Springer.

Ferreira, L., Pilastri, A., Martins, C. M., Pires, P. M., and Cortez, P. (2021). A comparison of automl tools for machine learning, deep learning and xgboost. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Geurts, P. (2001). Pattern extraction for time series classification. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 115–127. Springer.

Gijsbers, P., LeDell, E., Poirier, S., Thomas, J., Bischl, B., and Vanschoren, J. (2019). An open source automl benchmark. *arXiv preprint arXiv:1907.00909 [cs.LG]*. Accepted at AutoML Workshop at ICML 2019.

Gomez-Pilar, J., Gutiérrez-Tobal, G. C., Poza, J., Fogel, S., Doyon, J., Northoff, G., and Hornero, R. (2021). Spectral and temporal characterization of sleep spindles—methodological implications. *Journal of Neural Engineering*, 18(3):036014.

Gonçalves, I., Silva, S., Melo, J. B., and Carreiras, J. (2012). Random sampling technique for overfitting control in genetic programming. In *European Conference on Genetic Programming*, pages 218–229. Springer.

Guo, L., Rivero, D., Dorado, J., Munteanu, C. R., and Pazos, A. (2011). Automatic feature extraction using genetic programming: An application to epileptic eeg classification. *Expert Systems with Applications*, 38(8):10425–10436.

Hazarika, N., Chen, J. Z., Tsoi, A. C., and Sergejew, A. (1997). Classification of eeg signals using the wavelet transform. *Signal processing*, 59(1):61–72.

Hughes, G. (1968). On the mean accuracy of statistical pattern recognizers. *IEEE transactions on information theory*, 14(1):55–63.

Hutter, F., Caruana, R., Bardenet, R., Bilenko, M., Guyon, I., Kegl, B., and Larochelle, H. (2014). Automl 2014@ icml. In *AutoML 2014 Workshop@ ICML*.

Hutter, F., Kotthoff, L., and Vanschoren, J. (2019). *Automated Machine Learning*. Springer.

Ilyas, M. Z., Saad, P., and Ahmad, M. I. (2015). A survey of analysis and classification of eeg signals for brain-computer interfaces. In *2015 2nd International Conference on Biomedical Engineering (ICoBE)*, pages 1–6. IEEE.

Iranmanesh, S. and Rodriguez-Villegas, E. (2017). An ultralow-power sleep spindle detection system on chip. *IEEE transactions on biomedical circuits and systems*, 11(4):858–866.

Kevric, J. and Subasi, A. (2017). Comparison of signal decomposition methods in classification of eeg signals for motor-imagery bci system. *Biomedical Signal Processing and Control*, 31:398–406.

Koza, J. R. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and computing*, 4(2):87–112.

Lachner-Piza, D., Epitashvili, N., Schulze-Bonhage, A., Stieglitz, T., Jacobs, J., and Dümpelmann, M. (2018). A single channel sleep-spindle detector based on multivariate classification of eeg epochs: Mussdet. *Journal of neuroscience methods*, 297:31–43.

LeDell, E. and Poirier, S. (2020). H2O AutoML: Scalable automatic machine learning. *7th ICML Workshop on Automated Machine Learning (AutoML)*.

Lubba, C. H., Sethi, S. S., Knaute, P., Schultz, S. R., Fulcher, B. D., and Jones, N. S. (2019). catch22: Canonical time-series characteristics. *Data Mining and Knowledge Discovery*, 33(6):1821–1852.

Mei, N., Grossberg, M. D., Ng, K., Navarro, K. T., and Ellmore, T. M. (2017). Identifying sleep spindles with multichannel eeg and classification optimization. *Computers in biology and medicine*, 89:441–453.

Miranda, I. M., Aranha, C., de Carvalho, A. P. L., and Garcia, L. P. F. (2022). Genetic programming-based automl for eeg signal classification - a comparative study. In *Anais do X Symposium on Knowledge Discovery, Mining and Learning*. SBC.

Miranda, Í. M., Aranha, C., and Ladeira, M. (2019). Classification of eeg signals using genetic programming for feature construction. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1275–1283.

Motamedi-Fakhr, S., Moshrefi-Torbati, M., Hill, M., Hill, C. M., and White, P. R. (2014). Signal processing techniques applied to human sleep eeg signals—a review. *Biomedical Signal Processing and Control*.

Niedermeyer, E. and Ribeiro, M. (2000). Considerations of nonconvulsive status epilepticus. *Clinical Electroencephalography*, 31(4):192–195.

Olson, R. S. and Moore, J. H. (2019). Tpot: A tree-based pipeline optimization tool for automating machine learning. In *Automated Machine Learning*, pages 151–160. Springer.

Olson, R. S., Urbanowicz, R. J., Andrews, P. C., Lavender, N. A., Moore, J. H., *et al.* (2016). Automating biomedical data science through tree-based pipeline optimization. In

European Conference on the Applications of Evolutionary Computation, pages 123–137. Springer.

O'Reilly, C. and Nielsen, T. (2014). Assessing eeg sleep spindle propagation. part 2: experimental characterization. *Journal of Neuroscience Methods*, 221:215–227.

Parekh, A., Selesnick, I. W., Rapoport, D. M., and Ayappa, I. (2015). Detection of k-complexes and sleep spindles (detoks) using sparse optimization. *Journal of neuroscience methods*, 251:37–46.

Patti, C. R., Chaparro-Vargas, R., and Cvetkovic, D. (2014). Automated sleep spindle detection using novel eeg features and mixture models. In *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 2221–2224. IEEE.

Patti, C. R., Shahrbabaki, S. S., Dissanayaka, C., and Cvetkovic, D. (2015). Application of random forest classifier for automatic sleep spindle detection. In *2015 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 1–4. IEEE.

Poli, R., Langdon, W. B., and McPhee, N. F. (2008). *A field guide to genetic programming*.

Purcell, S., Manoach, D., Demanuele, C., Cade, B., Mariani, S., Cox, R., Panagiotaropoulou, G., Saxena, R., Pan, J., Smoller, J., *et al*. (2017). Characterizing sleep spindles in 11,630 individuals from the national sleep research resource. *Nature communications*, 8(1):1–16.

Shang, Z., Zgraggen, E., Buratti, B., Kossmann, F., Eichmann, P., Chung, Y., Binnig, C., Upfal, E., and Kraska, T. (2019). Democratizing data science through interactive curation of ml pipelines. In *Proceedings of the 2019 international conference on management of data*, pages 1171–1188.

Suchopárová, G. and Neruda, R. (2020). Genens: An automl system for ensemble optimization based on developmental genetic programming. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 631–638. IEEE.

Tran, B., Xue, B., and Zhang, M. (2016). Genetic programming for feature construction and selection in classification on high-dimensional data. *Memetic Computing*, 8(1):3–15.

Tsanas, A. and Clifford, G. D. (2015). Stage-independent, single lead eeg sleep spindle detection using the continuous wavelet transform and local weighted smoothing. *Frontiers in human neuroscience*, 9:181.

Tuggener, L., Amirian, M., Rombach, K., Lörwald, S., Varlet, A., Westermann, C., and Stadelmann, T. (2019). Automated machine learning in practice: state of the art and recent results. In *2019 6th Swiss Conference on Data Science (SDS)*, pages 31–36. IEEE.

Unser, M. and Aldroubi, A. (1996). A review of wavelets in biomedical applications. *Proceedings of the IEEE*, 84(4):626–638.

Vanneschi, L., Castelli, M., and Silva, S. (2010). Measuring bloat, overfitting and functional complexity in genetic programming. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 877–884.

Wang, H., Ma, C., and Zhou, L. (2009). A brief review of machine learning and its application. In *2009 international conference on information engineering and computer science*, pages 1–4. IEEE.

Weiner, O. M. and Dang-Vu, T. T. (2016). Spindle oscillations in sleep disorders: a systematic review. *Neural plasticity*, 2016.

Xanthopoulos, I., Tsamardinos, I., Christophides, V., Simon, E., and Salinger, A. (2020). Putting the human back in the automl loop. In *EDBT/ICDT Workshops*.

Xin, D., Wu, E. Y., Lee, D. J.-L., Salehi, N., and Parameswaran, A. (2021). Whither automl? understanding the role of automation in machine learning workflows. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–16.

Yamazaki, M., Tucker, D. M., Fujimoto, A., Yamazoe, T., Okanishi, T., Yokota, T., Enoki, H., and Yamamoto, T. (2012). Comparison of dense array eeg with simultaneous intracranial eeg for interictal spike detection and localization. *Epilepsy research*, 98(2-3):166–173.

Yang, F., Elmer, J., and Zadorozhny, V. I. (2021). Smartprognosis: Automatic ensemble classification for quantitative eeg analysis in patients resuscitated from cardiac arrest. *Knowledge-Based Systems*, 212:106579.

Zhuang, X., Li, Y., and Peng, N. (2016). Enhanced automatic sleep spindle detection: a sliding window-based wavelet analysis and comparison using a proposal assessment method. In *Applied Informatics*, volume 3. SpringerOpen.

Zöller, M.-A. and Huber, M. F. (2021). Benchmark and survey of automated machine learning frameworks. *Journal of Artificial Intelligence Research*, 70:409–472.