Proceedings of the Second International Conference on Innovative Mechanisms for Industry Applications (ICIMIA 2020)
IEEE Xplore Part Number: CFP20K58-ART; ISBN: 978-1-7281-4167-1

Automated Machine Learning: The New Wave of Machine Learning

Karansingh Chauhan[1], Shreena Jani[1], Dhrumin Thakkar[1], Riddham Dave[1],Jitendra Bhatia[1], Sudeep Tanwar[2], Mohammad S. Obaidat, Fellow of IEEE and Fellow of SCS[3]

[1]Vishwakarma Government Engineering College
Gujarat Technological University, Ahmedabad, India

c2karansingh@gmail.com
janishreena28@gmail.com
dhruminthkkr@gmail.com
ridhamdave5@gmail.com
jitendrabbhatia@gmail.com

[2]Institute of Technology
Nirma University, Ahmedabad, India

sudeep.tanwar@nirmauni.ac.in

[3] University of Sharjah
Dean of College of Computing and Informatics, University of Sharjah, UAE, King Abdullah II School of IT, University of Jordan, Jordan, and University of Science Ming Chuan University, Taiwan

m.s.obaidat@ieee.org

***Abstract*—With the explosion in the use of machine learning in various domains, the need for an efficient pipeline for the development of machine learning models has never been more critical. However, the task of forming and training models largely remains traditional with a dependency on domain experts and time-consuming data manipulation operations, which impedes the development of machine learning models in both academia as well as industry. This demand advocates the new research era concerned with fitting machine learning models fully automatically i.e., AutoML. Automated Machine Learning(AutoML) is an end-to-end process that aims at automating this model development pipeline without any external assistance. First, we provide an insights of AutoML. Second, we delve into the individual segments in the AutoML pipeline and cover their approaches in brief. We also provide a case study on the industrial use and impact of AutoML with a focus on practical applicability in a business context. At last, we conclude with the open research issues, and future research directions.**

***Index Terms*—Automated Machine Learning, Artificial Intelligence Meta Learning, Hyperparameter Optimization**

I. Introduction

Data analysis is a powerful tool for learning insights on how to improve the decision making, business model and even products. This involves the construction and training of a machine learning model which faces several challenges due to lack of expert knowledge [1]. This challenges can be overcomed by using automated machine learning(AutoML) field. AutoML refers to the process of studying a traditional machine learning model development pipeline to segment it into modules and automate each of those to accelerate workflow. With the advent of deeper models, such as the ones used in image processing [2], Natural Language Processing [3], etc., there is an increasing need for tailored models that can be crafted for specific workloads. However, such specific models require immense resources such as high capacity memory, strong GPUs, domain experts to help during the development and long wait times during training. The task gets critical as there is not much work done for creating a formal framework for deciding model parameters without the need for trial and error. These nuances emphasized the need for AutoML where automation can reduce turnaround times and also increase the accuracy of the derived models by removing human errors. In recent years, several tools and models have been proposed in the domain of AutoML. Some of these focus on particular segments of AutoML such as feature engineering or model selection, whereas some models attempt to optimize the complete pipeline. These tools have matured enough to be able to compare with human experts on Kaggle competitions and at times have beat them as well, showcasing their veracity. There are wide variety of applications based on AutoML such as autonomic cloud computing [4] [5], Intelligent Vehicular networks, Block Chain [6],Software Defined Networking [7] [8], among others.

This paper aims at providing an overview of the advances seen in the realm of AutoML in recent years. We focus on individual aspects of AutoML and summarize the improvements achieved in recent years. The motivation of this paper stems from the unavailability of a compact study of the current state of AutoML. While we acknowledge the existence of other surveys [9] [10] [11], their motive is to either provide an in-depth understanding of a particular segment of AutoML, provide just an experimental comparison of various tools used or are fixated towards deep learning models. The primary contributions of this paper are threefold:

1) We segment the AutoML pipeline into parts and review the contributions in each of these segments.
2) We explore the various state-of-the-art tools currently available for AutoML and evaluate them.
3) We also incorporate the advancements seen in machine learning which seems to be overshadowed by deep learning in recent years.

The rest of the paper is organized as follows. Section I-A describes the problem definition of AutoML and covers the contributions in AutoML with each subsection reviewing a specific segment. Section II discusses the recent trends and advancements seen in the domain of AutoML. Section IV covers a case study of the use-case of AutoML for insurance. Section V concludes the paper and provides future directions for the work that needs to be done in AutoML.

A. Automated Machine Learning

An AutoML is the process of automating the end-to-end process of applying machine learning to real-world problems. The problem of AutoML is a combinational one, where any proposed algorithm is required to find a suitable combination of operations for each segment of the ML pipeline to minimize the errors. Mathematically, AutoML can be expressed as:

$$^{O_P}\mathrm{C}_{O_S}+2^{N}.^{G(f_i,f_j)}\mathrm{P}_{N_M}+\sum_{m'\in M}\sum_{r\in R}P(m',r|m)P(r+\gamma.v(m')) \tag{1}$$

where,
O_P is standard pre-defined operation set
O_S indicates operations selected by the algorithms
$G(f_i, f_j)$ specifies generator function for creating new features
N represents the number of selected features
N_M = Maximum features to be selected

The standard data pre-processing operations are well defined and discussed in sectionII-A. While a completely raw data collection cannot be processed with these standard operations, datasets are usually refined to some extent and can work with such operations well. The automation in data pre-processing is defined as a series of actions that are selected(O_S) from the standard pre-defined operation set(O_P) and performed on the dataset. Feature Engineering is performed by selecting relevant features(2^N) from the dataset by finding dependant pairs((f_i, f_j)) and using them for generating new features($G(f_i, f_j)$). Model selection and hyperparameter optimization work on finding the optimal parametric configuration from an infinite search-space or from learning them (reinforcement learning) from previous models designed for various tasks. The final term in equation 1 demonstrates the probabilistic reinforcement learning used in recent years for constraining the configuration space.

The solution-space explosion due to exponentials and factorials, as shown in equation 1 is the core issue of AutoML. This explosion causes a high expense computationally and voids any accuracy advantage over humans. To address this problem, various research works that are proposed, allows a parameter configuration to granularly adjust the volume of the search space explored by any algorithm. Some works have removed the combination configurations deemed ineffective based on previous experience. There are lots of frameworks available for AutoML, which are shown in Table I.

II. Related work

This section describes the various segments of AutoML as per the taxonomy shown in Figure II. We present the most notable contributions seen in the domain of AutoML. We compare the various approaches adopted for each individual segment of AutoML.

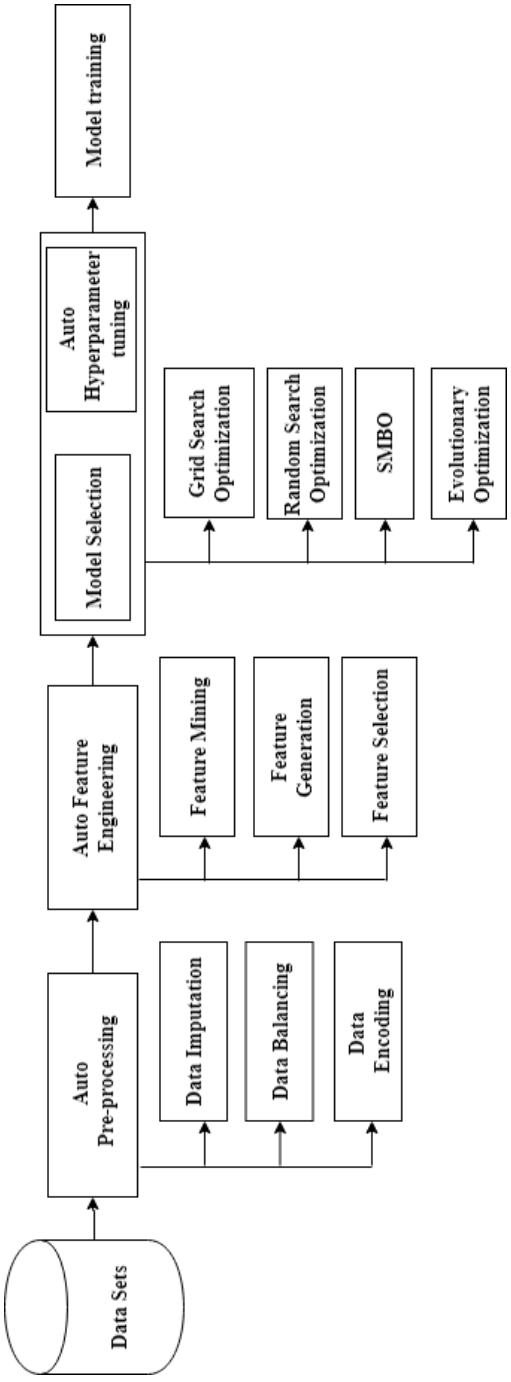

Fig. 1. Taxonomy of AutoML

TABLE I
SUMMARY OF THE AUTOML FRAMEWORKS

AutoML Tools	Data source		Preprocessing	FE	ML Task		Model Selection and HPO Techniques	NAS	Meta-Learning	UI	Authorization
	Structured	Unstructured			Supervised	Unsupervised					
H2O-AutoML	✓	✗	✓	✓	✓	✗	Random search and Grid search	✗	✓	✓	Open Source
DataRobot	✓	✓	✓	✓	✓	✓	Random search, Bayesian optimization	✓	✗	✓	Proprietary
Cloud AutoML	✓	✓	✓	✓	✓	✓	Genetic algorithm, Random search, Bayesian optimization	✓	✓	✓	Proprietary
TPOT	✓	✗	✗	✓	✓	✗	Genetic algorithm	✗	✗	✗	Open Source
Auto-Keras	✓	✓	✗	✓	✓	✗	Random search, Bayesian optimization	✓	✗	✗	Open Source
Auto-Weka	✓	✗	✗	✓	✓	✗	Random search, Bayesian optimization	✗	✓	✓	Open Source
ML BOX	✓	✗	✓	✓	✓	✗	SMAC Bayesian optimization	✗	✓	✓	Open Source
AutoSklearn	✓	✗	✗	✓	✓	✓	Random search, Bayesian optimization	✗	✓	✗	Open Source
Auto-Pytorch	✓	✗	✓	✓	✓	✓	Multi-fidelity optimization, Bayesian optimization (BOHM)	✓	✓	✗	Open Source

A. Data preprocesing

Data pre-processing guarantees the delivery of quality data derived from the original dataset. It is an important step due to the unavailability of quality data as a large portion of information generated and stored is usually semi-structural or even non-structured in form. However, even though it is a crucial part of any machine learning pipeline, it is reported to be the least enjoyable part, with authors [12] [13] stating that 60-80% of data scientists finding it to be the most mundane and tedious job.

In AutoML, certain data-preprocessing operations are hard-coded, which are then applied to a given dataset in certain combinations such that the overall clarity and usability of the data increases. [14] We have largely classified these operations into the following categories based on our surveys of recent papers as seen in Figure II-A.

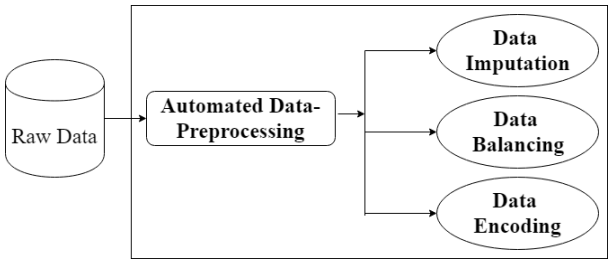

Fig. 2. Data pre-processing pipeline

1) Data Imputation:: Often datasets, in reality, may contain missing values for some different reasons (human error or unavailability of the data). There are two fundamental kinds of missing data as described by [15], which are missing at random (MAR) data and Missing completely at random (MCAR) data. The randomness of MCAR data is high enough that there is no overall bias towards any particular class, unlike MAR data, which are responsible for causing an increase in bias. In data Imputation, we deal with inconsistencies such as NaNs, spaces, Null values, incorrect data types, etc. This is addressed by replacing these values with multiple methods such as default value selection in which every problematic value is removed, and a pre-selected value takes its place. Another approach is to use the mean or median of the dataset column [16] to replace any missing value. Some approaches as regression imputation have used standard Deviation and Variance to compute the replacement [17] value for a given data column. Some data imputation technique with lighter time constraints uses the successive halving approach in Auto-WEKA [18]. XGboost algorithm [19] is also used widely in TPOT tool [20], and Auto-WEKA [21] for data imputation.

2) Data Balancing:: Data imbalance is a condition when one or more classes in a categorical dataset have higher observations than the rest of the classes. Feeding such imbalanced data leads [22] to the input majority class have an unjustified bias. The sample handling approach for data balancing will preprocess the training set to minimize class differences, and this issue can be resolved. Two techniques ordinarily utilized to tackle data imbalance are under-sampling and down-sampling [23] in which data is repeated or removed pairwise to maintain class balance across the dataset. Another approach to boost classifier performance in case of an imbalance is Ensemble learning derived from Beriman's work [22] in which the existing dataset is augmented to generate more data points in an attempt to increase training data. An approach for tackling data imbalance is to use cost-sensitive learning [24] instead of changing the dataset. Cost-sensitive learning uses the variable cost of misclassification to balance the bias of an imbalance class. It is suitable for a highly skewed dataset where certain classes are minorities. In the case of AutoML, tools such as TPOT [25] provide an implementation in its API to adjust for class-specific sensitivity to adjust for skewed classes.

3) Data Encoding:: To make the data human-readable, the training data is often labelled in words. Data Encoding refers to converting the provided feature labels into numerical form to allow computer machines to interpret them. Some of the common forms of data encoding are ordinal, one-hot, binary, hashing, target encoding, etc. Target encoding is the process of replacing a categorical value with the mean/median/mode of the target variable. While other label encoding assigns incremental values or binary columns to every label, the values assigned to them are not representative of any property of the given data. Target encoding assigns a meaningful label number which represents a certain property of the data such as the fraction of true values in the target variable. H2O.AI is an autoML tool that makes use of target encoding in its API. Auto-Sklearn uses one-hot for data encoding [25].

B. Feature Engineering

Features obtained from a dataset are seldom used as-is. We generally perform some operations to generate new features that are well-suited for a given problem. For example, in a housing dataset, we may want to combine the length and breadth of the house property to compute its area, which is a better feature and is regularly observed in the real-estate domain. However, the task of the creation of new features from existing features is an artful and domain-specific process. AutoML deals with this in three distinct segments

1) Feature Mining:: The dataset generated after pre-processing contains features, some of them which are useful for the model training, whereas others have minimal contribution towards the training phase of the machine learning model. Feature mining is responsible for picking out the impactful features from a given dataset. This is done by computing the relevant feature pairs. The measure of relevance is usually defined as the information gain or by measuring the relationship between any featured pair. AutoLearn [26] uses a cosine-transform and measures the euclidean distance on these transforms to determine the feature pairs which correlate.

2) Feature Generation:: Feature generation is a process of combining pre-existing features to generate new features. AutoLearn [26] achieves this by performing ridged regression over feature pairs to map the relationships and considers the

newly generated mapping as a relationship. COGNITO [27] uses a series of standard operations over a feature tree to generate new features as seen in Fig. II-B2. LFE(Learning Feature Engineering) [28] improves on COGNITO by learning over the previous datasets to learn the relationship between features and the transforms, which generated better accuracy outcomes. These transforms are evaluated for a given dataset by LFE to determine the operations which will lead to an increase in the performance of the machine learning model. Reinforcement learning has also been used by Khurana *et al.* [29] for generating features. They leveraged the feature tree structure used in COGNITO and incorporated a traversal policy to optimize transform exploration. Reinforcement learning encourages the exploration of transforms that are beneficial to the overall model and also applies a budget constraint. This constraint is needed to prevent the algorithm from performing an exhaustive search over the feature graph.

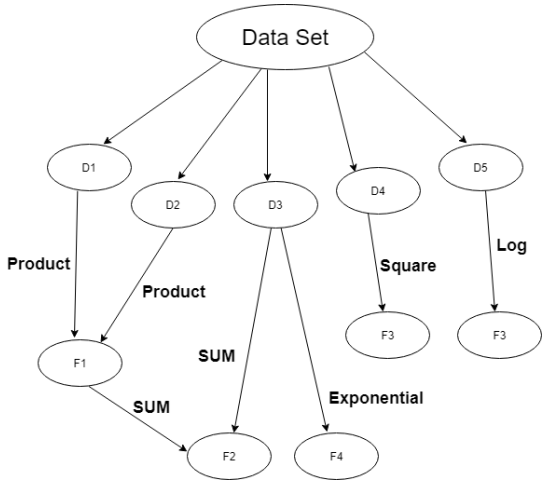

Fig. 3. An example of the feature tree generated by [27]

3) Feature Selection:: Feature Generation is an iterative process that leads to an explosion of total features. This is controlled by the selection phase based on the impact of a particular feature on the overall accuracy of the model. This is usually measured either by using a rank function or by measuring the loss of the model when a particular feature is excluded and included. ExploreKit [30] uses a novel ranking function based on a meta-feature classifier to determine which generated features are important. OneBM [31] utilizes the chi-square hypothesis to determine the features most relevant to the performance of the machine learning model. Information gain has also been used as a parameter for feature selection [26]. The above-described methods set a threshold to select only the most relevant features generated in an autoML pipeline.

C. Model Selection and Hyperparameter Optimization

The core of any machine learning pipeline is the model used to perform the prediction task. However, a single problem may have multiple model configurations with varying accuracy to tackle it. Hence, it is crucial to determine the most appropriate model keeping in mind the accuracy-execution time tradeoff. Conventionally, domain experts with previous experience approximate a model to be used. This manual task by humans follows an iterative trial and error approach for determining the model to be used, as shown in Fig. II-C. Once a model is finalized, the hyperparameter optimization is again performed manually to generate the final model. AutoML automates these above steps to reduce human dependency.

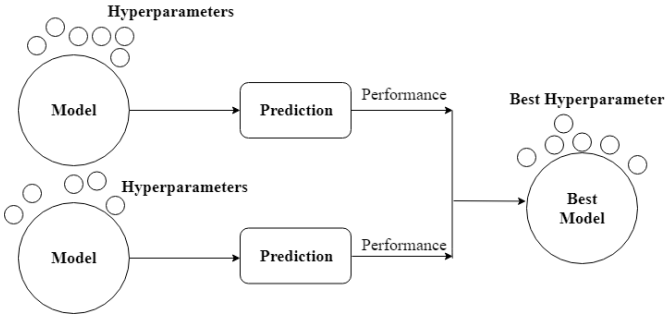

Fig. 4. Hyperparameter optimization by trial and error

Most of the AutoML tools and methods combine the problem of model selection and hyperparameter optimization into a single problem called the CASH(Combined Algorithm Selection and Hyperparameter) problem. CASH problem considers model selection and hyperparameters optimization as a single hierarchical hyperparameter optimization problem. At the root level, a hyperparameter resides, which selects between different learning algorithms or models. At the next level, model-specific hyperparameters are placed which are optimized to generate the final model. Auto-WEKA [21] and SmartML [32] are some of the tools, which consider model selection and hyperparameter optimization as a singular problem. The following approaches address CASH problem:

1) Grid search:: The Grid search was proposed as a traditional approach for the systematic exploration of the hyperparameter configuration space. It is simply a brute-force algorithm that searches through a pre-specified subset of hyperparameter space of the specific learning algorithm. The algorithm can be parallelized across multiple models with different configurations to accelerate the search. However, due to its brute-forcing characteristics, it is a very costly approach as for N different hyperparameters each having just two possible values, we have a total of 2n possible configurations [33].

2) Random Search:: To alleviate the exhaustive enumeration of combinations in a grid search, Random Search chooses random values from the hyperparameter subset independently. By navigating the grid of hyperparameters randomly, one can obtain a similar performance as a full grid search. However, this approach is surprisingly easy and effective. It is also well suited for gradient-free functions with many local minima [34]. Random search can outperform Grid Search in a scenario where the small number of hyperparameters affects the final

performance of the model [35].

3) Sequential Model-Based Optimization:: Random Search and Grid search performs hyperparameter checking independ of each other and often end up performing repeated and wasteful computations. To improve on their shortcomings, Sequential Model-based Optimization(SMBO) [36] was proposed, which uses a combination of regression and Bayesian optimization to select hyperparameters. It sequentially applies the hyperparameters and adjusts their values based on the Bayesian heatmap, which is a probabilistic distribution, as shown in Figure II-C3. The probabilistic approach of SMBO resolves the scalability issues that were rampant in grid search and random search.

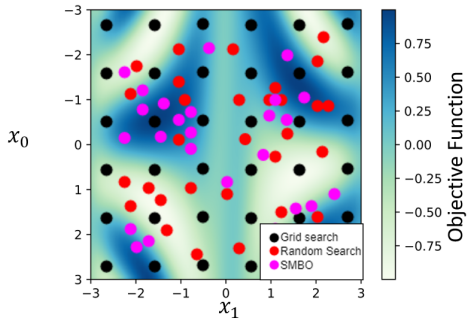

Fig. 5. A comparative example of hyperparameter selection behaviour of various strategies. Notice the selection clustering in the case of SMBO near the high scoring regions

To further improve this Bayesian optimization approach, the work in [37] introduces a deep neural network for global optimization of the hyperparameters. For neural networks, Mendoza *et al.* [38] introduced Auto-Net, an AutoML tool based on Bayesian optimization for tuning neural networks. The tool uses Stochastic Gradient Descent (SGD) as its optimizer for Hyperparameter optimization. The authors have also demonstrated a combined approach of using Auto-Net and Auto-SKLearn to outperform human adversaries by a significant margin of 10% in the AutoML Challenge 2018 [18].

The authors in [39] further improve upon the previous approaches which can be generalized across datasets using a transfer learning strategy. They achieve this by constructing a common hyperparameter surface of the previous hyperparameter selection plane and the target models hyperparameters. SmartML [32] is a meta-learning based framework for automated hyperparameter tuning and selection of ML models. It continuously learns from a given dataset and stores information about the meta-features of all the previously processed datasets to increase performance.

4) Evolutionary optimization:: Evolutionary optimization is inspired by biological evolution which follows Survival of the fittest. Such algorithms work by generating random agents, which perform a particular task and are scored on their performance. The agents are evaluated, and a breeding algorithm generates new child agents derived from the best agents in the former generation. This new generation again performs the same given task, and the cycle continues, as shown in Figure II-C4. For AutoML, evolutionary algorithms are used for tackling hyperparameter optimization by searching the configuration space for a given model.

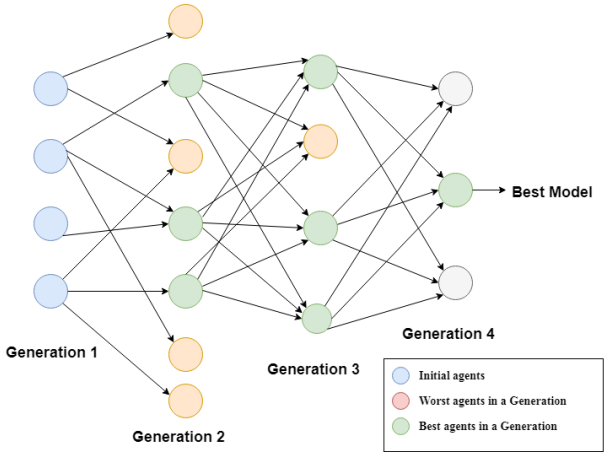

Fig. 6. An example of the evolutionary algorithm where every generation of best agents breed to generate next generation

To further imp Tree-based Pipeline Optimization Tool(TPOT) [40], an AutoML tool, makes use of evolutionary algorithms (genetic programming in particular) and has demonstrated its effectiveness on simulated and real-world genetic datasets. Autostacker [41] is another AutoML architecture that uses evolutionary models to optimize hyperparameter search. It produces candidate pipelines in each generation and evolves itself.

III. Discussion

Even though data pre-processing consumes a large chunk of time in an ML pipeline, it is astonishing to see the inadequate amount of work done to automate it. For data preprocessing, it can be noted that while the existing approaches are adequate for structured and semi-structured data, work still needs to be done to assimilate unstructured data. We suggest the incorporation of data-mining methods as they can deal with such unformed data. This can allow AutoML pipelines to create models capable of learning from Internet sources. In feature engineering, it should be noted that most methods used until now adhere to supervised learning. However, dataset specificity is high, and therefore, AutoML pipelines should be as generic as possible to accommodate the diverse datasets. Therefore, a gradual paradigm shift towards unsupervised learning is required to increase the ability of AutoML. To replace domain experts, feature generation should be able to work flexibly(such as the introduction of non-standard transforms) with the original feature sets. Reinforcement learning is a step in the right direction and needs to be inculcated further with feature engineering. Hyperparameter optimization has seen large improvements over the years, especially with the introduction of Bayesian optimization strategies such as

SMBO. However, the use of a continuously integrating meta-learning framework needs to be researched as its performance gain is high. Transfer learning has also been successfully used in the context of AutoML to show promising results. With the increase in the availability of task-specific pre-trained models, it should be expected to see an increase in the usage of transfer learning.

IV. Case Study: Impact of AutoML at GNP Seguros, an insurance company

The insurance industry usually prefers a data-driven approach to solve business problems. The multi-source data, generated in a massive amount, provoked the need for machine learning for further analysis and predictions. The significant challenges faced by the industry are the detection of false claims, utilization of unstructured data collected by the marketing and sales team, automation of transaction and claim processing, personalization of solutions for customers, among others.

GNP(Grupo Nacional Provincial) is one of the largest insurance companies in Mexico. Like any large and well-established company, GNP is undergoing a profound transformation for modernizing information systems and operations. To achieve this, the company is utilizing the cloud resources to centralize the generalized computations [42]. GNP is making significant efforts to organize and utilize all the operational information of the company in the central Data Lake [43] [44]. To extract value from Data lake, the company has begun to apply machine learning for getting intuitions as well as predicting and improving the company's performance based on their domain-specific factors [45]. For such a data-driven approach, a team of highly trained data scientists is required, which is financially taxing. In the earlier stages, the company's data scientists built and trained various models manually and thus achieved moderate accuracy for the prediction problem. To improve accuracy and reduce the amount of time and expenses, GNP adopted the tool called AutoML Tables provided by Google Cloud to simplify and speed up the creation of ML models and migrate the scarcity of highly trained data scientists. The company utilizes the provided tool to solve problems like Car claim risk, Detection of fraudulent healthcare claims, and Gender Labeling, which are discussed in detail below.

The car claim risk problem is defined as the task to predict the probability of the car having an accident using the given features/characteristics of the insured car and the owner. The company spends about USD 550 million on car damage claims annually, hence predicting risk amount accurately is the primary intention of the company. To solve the prediction problem, the company trained the model, using the AutoML Tables tool, which utilized the 21 features columns and over 1.34 million rows of raw data. The tool creates the model by selecting the most relevant features. By this, the accuracy of 98.1% has been achieved, which was much better than any of their previous manually trained models.

The detection of Healthcare fraudulent claims is one of the major problems faced by the company. Annually about 25000 false claims are filed for reimbursements, hence to reduce the loss of revenue due to such fraudulent activities, identification of false claims is required. The company utilized the AutoML Tables tool to train the model using features like patient history, hospitals, specific disease of patient, invoices, agent who sold the policy, etc. The tool is responsible for the optimal algorithm selection, pre-processing and feature selection. Based on the data provided, an appropriate anomaly detection algorithm is selected, and its hyperparameters are tuned accordingly. Using such a model, an accuracy of 96.64% is achieved for false claim detection. This model outperforms the existing in-house fraud detection model by 20% to 30%.

The labelling of gender can also be considered as one of the problems for the company. GNP not only provides insurance to the individual, but also for a collective group of people, for example, complete family or a company's employee network. In a general scenario, for the group insurance, the sales team provides data in CSV(Comma Separated Values) format to the underwriting department. For such a collective insurance policy, gender value in the data is the utmost requirement. However, missing values are frequently encountered in the columns representing gender features; hence, the identification of gender based on the persons name is required. To learn the gender of the person based on his/her corresponding full name, the company utilizes an AutoML model trained by the Tables tool. The AutoML tool selected the best suitable classification algorithm for the task and optimally tuned the hyperparameters accordingly to achieve an accuracy of 99.2%.

In this particular case, a single AutoML tool was able to tackle three problems head-on and created machine learning models for the same with minimal human intervention. This showcases the need as well as the opportunities the domain of AutoML provides, especially in the business sector.

V. Conclusion and Future Directions

In this paper, we provide insights to the readers about the various segments of AutoML with a conceptual perspective. Each of these segments has various approaches that have been briefly explained to provide a concise overview. We also discuss the various trends seen in recent years including suggestions of thirsty research areas which need attention. We also put forward some future directions that can be explored to extend the research in the domain of AutoML. We suggest that the research exploration can be done in the direction of a generalized AutoML pipeline, which can accept datasets of a wide range and a central meta-learning framework be established that acts as a central brain for approximating the pipelines for all future problems statements.

References

[1] Lukas Tuggener, Mohammadreza Amirian, Katharina Rombach, Stefan Lörwald, Anastasia Varlet, Christian Westermann, and Thilo Stadelmann. 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, 2019.

[2] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] Avatar Jaykrushna, Pathik Patel, Harshal Trivedi, and Jitendra Bhatia. Linear regression assisted prediction based load balancer for cloud computing. In *2018 IEEE Punecon*, pages 1–3. IEEE.

[5] Jitendra Bhatia, Ruchi Mehta, and Madhuri Bhavsar. Variants of software defined network (sdn) based load balancing in cloud computing: A quick review. In *International Conference on Future Internet Technologies and Trends*, pages 164–173. Springer, 2017.

[6] Ishan Mistry, Sudeep Tanwar, Sudhanshu Tyagi, and Neeraj Kumar. Blockchain for 5g-enabled iot for industrial automation: A systematic review, solutions, and challenges. *Mechanical Systems and Signal Processing*, 135:106382, 2020.

[7] Jitendra Bhatia, Yash Modi, Sudeep Tanwar, and Madhuri Bhavsar. Software defined vehicular networks: A comprehensive review. *International Journal of Communication Systems*, 32(12):e4005, 2019.

[8] Jitendra Bhatia, Ridham Dave, Heta Bhayani, Sudeep Tanwar, and Anand Nayyar. Sdn-based real-time urban traffic analysis in vanet environment. *Computer Communications*, 149:162 – 175, 2020.

[9] Xin He, Kaiyong Zhao, and Xiaowen Chu. Automl: A survey of the state-of-the-art. *arXiv preprint arXiv:1908.00709*, 2019.

[10] Radwa Elshawi, Mohamed Maher, and Sherif Sakr. Automated machine learning: State-of-the-art and open challenges. *arXiv preprint arXiv:1906.02287*, 2019.

[11] Anh Truong, Austin Walters, Jeremy Goodsitt, Keegan Hines, Bayan Bruss, and Reza Farivar. Towards automated machine learning: Evaluation and comparison of automl approaches and tools. *arXiv preprint arXiv:1908.05557*, 2019.

[12] Shichao Zhang, Chengqi Zhang, and Qiang Yang. Data preparation for data mining. *Applied artificial intelligence*, 17(5-6):375–381, 2003.

[13] Erhard Rahm and Hong Hai Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3–13, 2000.

[14] Dipali Shete and Sachin Bojewar. Auto approach for extracting relevant data using machine learning. *International Journal of Electronics*, 6:0, 2019.

[15] Carol M Musil, Camille B Warner, Piyanee Klainin Yobas, and Susan L Jones. A comparison of imputation techniques for handling missing data. *Western Journal of Nursing Research*, 24(7):815–829, 2002.

[16] RB Kline. Principles and practice of structural equation modeling. 1998. *New York: Guilford*, 1998.

[17] Joseph F Hair, Rolph E Anderson, Ronald L Tatham, and William C Black. Multivariate data analysis. englewood cliff. *New Jersey, USA*, 5(3):207–2019, 1998.

[18] Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. Practical automated machine learning for the automl challenge 2018. In *International Workshop on Automatic Machine Learning at ICML*, pages 1189–1232, 2018.

[19] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[20] Tpot: Skewed classes. https://github.com/EpistasisLab/tpot/blob/v0.9.5/tpot/metrics.py. (Accessed: September 10, 2019).

[21] Chris Thornton, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 847–855. ACM, 2013.

[22] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[23] Chris Drummond, Robert C Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, volume 11, pages 1–8. Citeseer, 2003.

[24] Mohamed Bekkar and Taklit Akrouf Alitouche. Imbalanced data learning approaches.

[25] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. Auto-sklearn: Efficient and robust automated machine learning. In *Automated Machine Learning*, pages 113–134. Springer, 2019.

[26] Ambika Kaul, Saket Maheshwary, and Vikram Pudi. Autolearnautomated feature generation and selection. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 217–226. IEEE, 2017.

[27] Udayan Khurana, Deepak Turaga, Horst Samulowitz, and Srinivasan Parthasrathy. Cognito: Automated feature engineering for supervised learning. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, pages 1304–1307. IEEE, 2016.

[28] Fatemeh Nargesian, Horst Samulowitz, Udayan Khurana, Elias B Khalil, and Deepak S Turaga. Learning feature engineering for classification. In *IJCAI*, pages 2529–2535, 2017.

[29] Udayan Khurana, Horst Samulowitz, and Deepak Turaga. Feature engineering for predictive modeling using reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[30] Gilad Katz, Eui Chul Richard Shin, and Dawn Song. Explorekit: Automatic feature generation and selection. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 979–984. IEEE, 2016.

[31] Hoang Thanh Lam, Johann-Michael Thiebaut, Mathieu Sinn, Bei Chen, Tiep Mai, and Oznur Alkan. One button machine for automating feature engineering in relational databases. *arXiv preprint arXiv:1706.00327*, 2017.

[32] Mohamed Maher and Sherif Sakr. Smartml: A meta learning-based framework for automated selection and hyperparameter tuning for machine learning algorithms. In *EDBT: 22nd International Conference on Extending Database Technology*, 2019.

[33] Steven M LaValle, Michael S Branicky, and Stephen R Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *The International Journal of Robotics Research*, 23(7-8):673–692, 2004.

[34] Francisco J Solis and Roger J-B Wets. Minimization by random search techniques. *Mathematics of operations research*, 6(1):19–30, 1981.

[35] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[36] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization*, pages 507–523. Springer, 2011.

[37] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *International conference on machine learning*, pages 2171–2180, 2015.

[38] Hector Mendoza, Aaron Klein, Matthias Feurer, Jost Tobias Springenberg, and Frank Hutter. Towards automatically-tuned neural networks. In *Workshop on Automatic Machine Learning*, pages 58–65, 2016.

[39] Dani Yogatama and Gideon Mann. Efficient transfer learning method for automatic hyperparameter tuning. In *Artificial intelligence and statistics*, pages 1077–1085, 2014.

[40] Randal S Olson and Jason H Moore. Tpot: A tree-based pipeline optimization tool for automating machine learning. In *Automated Machine Learning*, pages 151–160. Springer, 2019.

[41] Boyuan Chen, Harvey Wu, Warren Mo, Ishanu Chattopadhyay, and Hod Lipson. Autostacker: A compositional evolutionary learning system. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 402–409. ACM, 2018.

[42] Jitendra Bhatia and Malaram Kumhar. Perspective study on load balancing paradigms in cloud computing. *IJCSC*, 6(1):112–120, 2015.

[43] Natalia Miloslavskaya and Alexander Tolstoy. Big data, fast data and data lake concepts. *Procedia Computer Science*, 88:300–305, 2016.

[44] Jitendra Bhagwandas Bhatia. A dynamic model for load balancing in cloud infrastructure. *Nirma University Journal of Engineering and Technology (NUJET)*, 4(1):15, 2015.

[45] Jai Prakash Verma, Sudeep Tanwar, Sanjay Garg, Ishit Gandhi, and Nikita H Bachani. Evaluation of pattern based customized approach for stock market trend prediction with big data and machine learning techniques. *International Journal of Business Analytics (IJBAN)*, 6(3):1–15, 2019.