Automated Feature Synthesis From Relational Database For Data Science Related Problems

Afifa Fatima, Faizan Ali Khan, Aneeq Raza, Abdul Basit Kamran
Computer Science
Sir Syed CASE Institute of Technology
Islamabad, Pakistan
afifafatima095@gmail.com, cs14.046.faizan@gmail.com, aneeqraza2013@gmail.com, basitabdul047@gmail.com

***Abstract*— Feature Synthesis is the most important aspect of Data Science. This process requires domain knowledge of the data to produce meaningful features that support decision-making algorithms. The biggest technical obstacle which a data scientist must subdue is the synthesis of such features that would assist AI algorithms to predict the target variable with the least error. Feature synthesis is a crucial step toward the application of machine learning which consumes time and resources. Automated Feature Synthesis obviates the tedious task of data scientists. Data Science Machine and One Button Machine are pioneering researches that explored the possibility of feature synthesis from raw relational data by applying linear exploratory algorithms which increase the dimensionality of the feature matrix exponentially, and hence, they consume a lot of time and memory. Our research revolves around the process of feature synthesis from the structured and unstructured data types for relational data which will provide an effective automation of a hectic manual task for the non-experts consuming a reasonable amount of resources. We devised a solution to reduce the time complexity of the operation by reducing dimensionality while considering the target variable by using approximate dynamic programming. The model is designed utilizing the concept of sparse reward setting, and it tries to achieve maximum reward as it explores the feature set. The target variable plays its part to compute the reward eventually leading to an optimum feature set. This process reduces data exploration time, cost and effort for both data scientists and non- experts.**

Keywords- Approximate dynamic programming, Automated Feature Synthesis

I. INTRODUCTION

Data which is voluminous, complex and relational in nature is provided by the clients to data scientists for extracting insights which would help them to devise new business strategies. In 2015, researchers at MIT presented the Deep Feature Synthesis algorithm which automated the process of feature engineering, and they competed in online data science competitions. DFS performs feature engineering on the structured, transactional and relational datasets [10]. Their work was followed by IBM's One Button Machine. One Button Machine (OneBM) is a framework that supports feature engineering from relational data, aiming at tackling the aforementioned challenges. to OneBM works directly with multiple raw tables in a database. It joins the tables

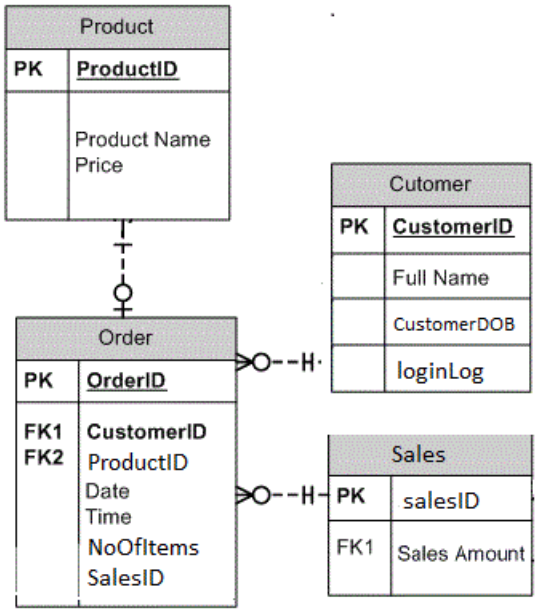

Figure 1. Entity diagram of a dataset of an e-commerce company

incrementally, following different paths on the relational graph. It automatically identifies data types of the joint results, including simple data types (numerical or categorical) and complex data types (set of numbers, set of categories, sequences, time series and texts). It applies corresponding pre-defined feature engineering techniques on the given types. The researchers at IBM state that feature engineering automation helps data scientists cut back information exploration time. It enables non-experts, who are not familiar with data science, to extract features from their data with a less effort, time and cost. . One BM used an Apache Spark cluster with 12 machines to run OneBM, where every machine has 92 GBs of memory and 12 cores [11].

For example, an online e-commerce website is maintaining all of its customer data. It also offers discounts from time to time. Black Friday is its one of the biggest sales of the year in which it targets its loyal customers who stays for a long period of time. The company bears loses due to one-time opportunity seekers. Such customers purchase products on Black Friday and then do shop for the next

978-1-5386-9355-1/18/$31.00 ©2018 IEEE
DOI 10.1109/FIT.2018.00020

whole year. The company comes up with following solution: The company identifies the customers that are likely to become loyal customers and target them through personalized sales. The entity diagram in Figure 1 shows the basic relational database of the company. The Customer table maintains the information of all the users, the Orders table contains the information of the customers and their orders. Customers and Orders table are having "one to many relationship". It means multiple entries of one single customer which may buy multiple products over the period of time may occur. Products and Orders entities are also having one to many relationships. One product_id can appear zero, one or many times in the Orders table. It is important to discuss the data types of few attributes in some of the entities.

(a) customer_id, product_id, number_of_items are numerical values (int/ long int)

(b) purchase_date, customer_dob and login_log are timestamps

(c) customer category is categorical data which represents 4 values which defines the types of their accounts on basis of their payments

(d) sale_item is of bool data which saves if the item was the sale item or not

The relational data of the company consists of various tables. The relational data (fig 1) that is provided by the company cannot be readily processed by the predictive algorithms. In order to process them, data scientists are required to perform various mathematical functions on each attribute and join data in such meaningful way that it conveys right information to the algorithms that are going to predict any target variable. The data scientists manually analyze the data, make use of human intuition and intelligence and transform the above attribute. In the above provided entities:

Customers:

(a) customer_dob is a timestamp which need to be converted into three categories as 1 = young, 2 = middle age, 3 = old and we can also break it into day, month and year.

(b) login_log can provide the information that how frequently the user uses the website. It can be converted into the categories as 1 = not frequent, 2 = normal, 3 = very frequently

(c) similarly, gender has two categories 1 = Male, 2 = Female

Orders:

purchase_date conveys the information about the customer purchases. It is a valuable information telling the history of a customer that how frequently the customer has bought an item. It should be converted into numerical data using group by and aggregate functions because it is valuable information and should not be transformed into few categories.

Rest attributes are going to be omitted or are already in the right form. It is important to join Customers, Orders and Sales table after applying the above transformations and then again look for important features that can be obtained from it.

The above mentioned manual process of transformation is termed as "Feature Synthesis" of raw data. Feature engineering is one of the most important and time consuming tasks in predictive analytics [10]. According to the participants of the Kaggle's competitions it is the most time and resource consuming process [12].

Class is decided by the data scientist and it depends on the requirement. This data can now be processed by predictive algorithms. The aim of our research is to automate the process of Feature Synthesis effectively through an algorithm. The example mentioned above utilizes the structured data. The process of automation should effectively work on various types of data. The data can be structured and unstructured. Unstructured data includes images, texts, series, sequences and so on.

TABLE I. FEATURE MATRIX

Feature Matrix			
customer_ID	***age_category***	***Sum_purchasedItem***	***Class/target_variable***
1	young	5	Not Loyal
2	old	8	Loyal
3	old	10	Loyal

We designed approximate dynamic programming based algorithm which generates features by considering the target variable beforehand. It is also a data science machine which performs feature selection and provides input to the predictive algorithms.

II. RELEATED WORK

Feature engineering or feature synthesis is a crucial step in the process of predictive modeling. It involves the transformation of given feature space, typically using mathematical functions, with the objective of reducing the modeling error for a given target. It involves domain knowledge, intuition, and most of all, a lengthy process of trial and error. In 2015, a paper (DSM) by MIT captured the attention of the wide audience, and it claimed to automate the hectic manual process of the data scientist. The name Deep Feature Synthesis comes from the algorithm's ability to stack transformation calculations to generate complex features. In each iteration when transform configuration has stacked the depth, d of a feature increases. The Deep Feature Synthesis traverses recursively along the relationships between data points in the dataset by applying mathematical functions over the data along the path to create final features. The final output is a multiplied base table that represents a much larger portion of the relational statistics. DSM identifies relationships between distinct entities, and it helps to gather new features. The connections between data points in a solitary dataset also help in deriving significant features.

The Data Science Machine and Deep Feature Synthesis algorithm were constructed on a MySQL database using the 'InnoDB' for tables. The raw datasets were manually converted to a MySQL schema for processing by the Data Science Machine. Basically, it is an end-to-end system for relational and structured data that automatically synthesizes features for machine [10]. A downside of the DSM framework that it does not support feature learning for unstructured non-numerical data such as sets, sequences, time-series or text. Features extracted by applying basic statistics functions. In many cases, there is a need to perform feature learning from the entire collected data and the target variable. For unstructured data, the features are beyond simple statistics. The focus is on important structure and patterns in the data [11]. DSM uses hyperparameter tuning to obtain convincing results, otherwise, the efficiency of the algorithms decreases drastically. One Button Machine, a framework that allows data scientists to perform feature learning on different kinds of structured and unstructured data. One Button Machine focuses primarily on preparing features from relational databases with multiple tables that can be used as input for predictive models. One Button Machine joins the tables incrementally and traverses the entity graph where nodes are tables(entities) and edges are relations (primary/foreign keys) between tables of a relational database. It has the capacity to automatically identify data types of the joint results and applies corresponding pre-defined feature engineering techniques on the given types. Aggregation statistics can be specified by user, or can be a generic aggregation function depending on the data type. One Button Machine applies sub sampling to deal with related entities and also uses dynamic caching of joined tables to reduce the memory consumption. OneBM, implemented in Apache Spark cluster, a framework for analyzing massive amounts of data. To run OneBM 12 machines are required, where every machine has 92 GBs of memory and 12 cores. In KDD cup 2014, OneBM outperformed DSM in terms of prediction accuracy and ranking on leader boards by improving its ranks on the private leader board from 145 to 80 [11]. One Button Machines claims to be a solution for non-experts but at the same time it fails to perform under restricted resources.

III. Problem Description

The process of feature synthesis involving structured and unstructured relational data utilizing inductive logic programming, feature optimization and selection is computationally extensive and intense task. Moreover, the possible dimension of the feature set is unbounded due to the iterative and relational nature of the data sets. Inclusion of each new feature requires validation and training which is computationally hectic process. The approaches mentioned in related work compute whole feature set, and then they select and validate the features using random forest. This process introduces redundant features which consume time and memory. The feature synthesis in sequential manner and training the model at the end after generating all the features restricts the new ideal features.

IV. Methodology

Our research proposes an exploration technique which enumerates the entity diagram of the data set and creates a stack of views having local features. A transformation on a view applies transformation functions on all the features or feature sets. Then on that single view, feature selection, training and validation is performed. Here the target variable plays its role. Validation and discarding irrelevant features at a view reduces the consumption of resources. The cross validation of all these views helps to select the view with the most valuable features. It also helps to avoid greedy approach because any time we lose the value of a view we can go back the some other view with more promising performance. This process utilizes the concept of dynamic programming presented by Richard Bellman. It is the process of breaking the problems into smaller sub problems recursively. Then choosing the optimal solution of the sub problem and evaluating the relationship between the sub problem and larger problem through Bellman equation.

ALGORITHM I. AUTOMATED FEATURE SYNTHESIS

```
v :  Local View
F : Global Feature Set
f : Local Feature Set
n : leafNode

Input: Dataset D0, TargetVariable y, Budget d, MainTable m,
transformPrimitives, Threshold t, S viewsSets

Output: View V having the features

G  <-  entityGraph(D0, m)
P  <- enumeratePaths(G, m)


while p in P

v, n <- createView(D0, d, p)
f <- transformPrimitives(v, n)
f <- featSelection(f, y)
score <- calculateScore(f, y)
s = {v, f, score}
if (score >= t)
S <-  S U s
 else
discard

V <- selectView(S)

Return V
```

A. Transform primitives

It is the set of predefined functions which contain the information about the operations that can be performed on different data types. For example, a attribute is of type

timestamp, the operations that are implemented in transformed configuration (table 4.1) will return seven or more new columns/attributes after performing operations on one column/attribute which may be useful as features for predictive algorithm. Lastly, machine learning algorithms like SVM or neural networks help in creating predictive model which utilizes the features that are obtained from the previous set of operations and predicts the target variable

B. Feature Selection

This module selects the features that are important to predict the target variable and removing the duplicate features. It is a multitude of decision trees at training time, outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct decision trees' habit of over fitting to their training set. We filtered out irrelevant features having low correlation and information gain at two levels.

V. EXPERIMENTS AND RESULTS

In this section we discuss the experimental results. OneBM is considered as a baseline to compare to in addition to manual feature engineering approaches provided by Kaggle participants. Three Kaggle competitions with complex relational graphs and different problem types (classification, regression and ranking) were chosen for validation Experimental settings and running time: A Google cloud computing machine with 8 cores, 50 GB of memory and 1TB of disk space. OneBM outperformed DFS/DSM and it utilized Apache Spark cluster with 12 machines, where every machine has 92 GBs of memory and 12 cores [11]. We compared our running times to OneBM and DFS in Table I. The results from Kaggle are reported in Table II.

TABLE II. COMPARASION OF RUNNING TIME

Data	♯ tables	♯ columns	size	OneBM	AFS	DFS
KDD Cup 2014	4	51	0.9 GB	12.9 h	6.8 h	NA
Coupon purchase	7	50	2.2 GB	2.8 h	1.6 h	84.04 hours
Grupo Bimbo	5	19	7.2 GB	56 min.	25 min	2 weeks (not finished)

TABLE III. KAGGLE RESULTS

Competition	Task	Metric	DFS	OneBM	AFS
KDD Cup 2014	Classification	AUC	0.586	0.622	0.619
Groupo Bimbo	Regression	LRMSE	NA	0.475	0.485
Coupon Purchase	Ranking	LRMSE	0.005635	**0.007416**	0.004228

VI. CONCLUSION AND FUTURE WORK

We have shown that feature synthesis for relational data can be automated using predefined sets of transform primitives in AFS algorithm. We demonstrated the expressiveness of the generated features on 3 datasets from different domains. We conducted the whole process without human involvement by enabling our algorithm to generalize to different datasets. Overall, the system is competitive with human solutions, DSM and OneBM for the datasets we tested on. This opens many interesting research directions for the future. For example, the AFS in this work is not auto-tuned due to the efficiency issue. Future work could focus on efficient methods for hyper parameter tuning to boost the current results even more.

REFERENCES

[1] A. Biem, M. Butrico, M. Feblowitz, T. Klinger, Y. Malitsky, K. Ng, A. Perer, C. Reddy, A. Riabov, H. Samulowitz, D. M. Sow, G. Tesauro, and D. S. Turaga. Towards cognitive automation of data science. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA., pages 4268– 4269, 2015.

[2] E. Brochu, V. M. Cora, and N. De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv: 1012.2599, 2010.

[3] L. Getoor and B. Taskar. Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning). The MIT Press, 2007.

[4] U. Khurana, D. Turaga, H. Samulowitz, and S. Parthasarathy, editors. Cognito: Automated Feature Engineering for Supervised Learning , ICDM 2016.

[5] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown. Autoweka 2.0: Automatic model selection and hyperparameter optimization in weka. Journal of Machine Learning Research, 17:1–5, 2016.

[6] R. S. Olson, N. Bartley, R. J. Urbanowicz, and J. H. Moore. Evaluation of a tree-based pipeline optimization tool for automating data science. In Proceedings of the Genetic and Evolutionary Computation Conference 2016, GECCO '16, pages 485–492, New York, NY, USA, 2016. ACM.

[7] Dhar, V. (2013). "Data science and prediction". Communications of the ACM. 56(12): 64

[8] Jeff Leek (2013-12-12). "The key word in "Data Science" is not Data, it is Science". Simply Statistics.

[9] Hayashi, Chikio (1998-01-01). "What is Data Science? Fundamental Concepts and a Heuristic Example". In Hayashi, Chikio; Yajima, Keiji; Bock, HansHermann; Ohsumi, Noboru; Tanaka, Yutaka; Baba, Yasumasa. Data Science, Classification, and Related Methods. Studies in Classification, Data Analysis, and Knowledge Organization. Springer Japan. pp. 40–51.

[10] J. M. Kanter and K. Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In Data Science and Advanced Analytics (DSAA), 2015

[11] Hoang Thanh Lam, Johann-Michael Thiebaut, Mathieu Sinn, Bei Chen, Tiep Mai, Oznur Alkan. One button machine for automating feature engineering in relational databases

[12] Kaggle.com. (2017). Kaggle: Your Home for Data Science. [online]Available at: https://www.kaggle.com/general/3761 [Accessed 8 Dec. 2017]

[13] Hastie, Trevor; Tibshirani, Robert; Friedman, Jerome (2008). The Elements of Statistical Learning (2nd ed.). Springer. ISBN 0-387-95284-5.

[14] Tripathy, V. (2013, 8 8). A Comparative Study Of Multi-Relational Decision. INTERNATIONAL JOURNAL OF SCIENTIFIC &TECHNOLOGY RESEARCH

[15] Leiva, H. A. (2002). MRDTL: A multi-relational decision tree learning algorithm

[16] Jing-Feng Guo, J. L.-F. (2007). An Efficient Relational Decision Tree Classification Algorithm . Third International Conference on Natural Computation

[17] Atramentov, A. (2003). Multi-relational decision tree algorithm.