Automated Feature Enhancement for Predictive Modeling using External Knowledge

Sainyam Galhotra[1] Udayan Khurana[2] Oktie Hassanzadeh[2] Kavitha Srinivas[2] Horst Samulowitz[2] Miao Qi[3]
[1]University of Massachusetts Amherst, [2]IBM Research, [3]RPI
sainyam@cs.umass.edu,{ukhurana, hassanzadeh, kavitha.srinivas, samulowitz}@us.ibm.com, qim@rpi.edu

***Abstract*—Supervised machine learning is the task of learning a function that maps features to a target. The strength of that function or the model depends directly on the features provided to the learning algorithm. Specifically, a crucial means of improving the model quality is to add new predictive features. This is often performed by domain specialists or data scientists. It is a hard and time-consuming task because the domain expert needs to identify data sources for new features, join them, and then select those that actually are relevant to the prediction. We present a new system called *KAFE (Knowledge Aided Feature Engineering)*, an interactive predictive modeling system that automatically utilizes structured knowledge present on the web to perform feature addition to improve the accuracy of predictive models. In this proposal, we describe the key techniques such as feature inference and selection, relevant data indexing, and demonstrate its use through an interactive Jupyter notebook.**

***Index Terms*—knowledge base, feature engineering**

I. Introduction

In recent years, we have witnessed proliferation of predictive modeling application in various domains. The cornerstone of building effective models is successful feature engineering. It is a lengthy and time-consuming process that relies on the domain knowledge of the data scientist, and often accounts for up to 80% of the time involved in building predictive models[1]. For instance, when building a predictive model of sales at a store, seasonal holidays such as Christmas are important to encode as features, because they account for surges in sales. Similarly, weather can dampen store sales at a brick and mortar store, and so could represent a key feature for predicting sales.

Often, the domain expertise needed for the addition of such features is not available or is prohibitively expensive to obtain. Moreover, even when it is available, the addition of features is mostly a human-driven process. It is tedious due to the guesswork, coding, and trial and error type of experiments involved. However, lack of investment in feature engineering impacts the quality of models and consequently, results in a lack of confidence in the use of such models in the real world. Automation of these time-consuming aspects of a machine learning cycle is therefore an area of significant importance and recent interest. Different approaches to perform automated feature engineering have been proposed recently and are being used in the industry[2]. Recent works on feature engineering or enhancement [4], [5], [7], [9] have proposed different methods for *transforming* the feature space using mathematical functions in order to morph the data into a more suitable feature set. Similarly, Deep Learning performs feature engineering implicitly by embedding features based on learned functional mappings. However, most of these techniques are oblivious to the semantic understanding of the features in data. Moreover, they do not consider additional information available from various external sources. Our focus is to fill that gap; specifically, we focus here on the problem of finding **new** data sets and semantically informative features relevant to the predictive model at hand, and adding those that have predictive value automatically into the model. We believe, this is the first work that considers addition of features having semantic explainable relationships with the input data and the techniques presented in this paper provide a valuable addition to the problem of automated feature engineering.

The web contains a plethora of expert knowledge in the form of encyclopedic data, books, blogs, dialogs, and more. A part of this information exists as structured knowledge, in the form of knowledge graphs, ontologies, web tables amongst other sources. Given the recent progress of building knowledge graphs for various domains to capture domain knowledge, devising a system to leverage this high quality structured information is of utmost importance. Our work demonstrates a system called KAFE that aims to utilize this information to improve classification and regression problems.

Given the gigantic scale of information present on the web coupled with heterogeneity of information available from different sources, automatic identification of relevant knowledge for predictive modeling is not trivial. This work demonstrates novel techniques to identify potential matches of the given data set with structured information on the web and quickly identifies the smallest set of features that help to boost the quality of a classifier. The demonstration provides a hands-on navigation through the various steps of our pipeline. It provides a handle to the end-user for better understanding of the performance of different components and the ability to plug in different techniques to monitor model performance.

Work done during first author's internship at IBM Research.
[1]https://tinyurl.com/yyb83ujh

[2]https://www.featuretools.com/, https://www.h2o.ai, https://www.ibm.com/cloud/watson-studio/autoai, https://cloud.google.com/automl-tables/

One of the major challenges of building our system was to index the information available from various data sources like DBpedia, Wikidata and other web tables (more than 20 million). The dual benefit of achieving fast lookup speed (less than 1 millisecond to identify one entity match) and knowledge inference from millions of data sources are some of the notable strengths of KAFE.

II. Related Work

Feature engineering has traditionally been a manual process. Recently, there has been some work to automate it. For instance, Cognito [7], [8] performs trial and error through reinforcement learning to find suitable "transformation functions" that generate new features from base features. Katz et al. [5] rely on a rather exhaustive algorithm that greedily evaluates many choices for feature transformations. LFE [9] learns patterns of associations between features, transforms and target from historical data to predict transforms for new data. All of these methods do not add new information that is external to the given data. A detailed account of such automated feature engineering techniques can be found here [6].

Friedman et al. [2] recursively generate new features using a knowledge base for text classification. FeGeLOD [12] extracts features from Linked Open Data via queries. These approaches, however, fail to thoroughly explore the available information from external web sources, and apply it to general machine learning problems. In contrast, our system automatically infers and selects novel and useful features from external sources such as web tables to boost the predictive performance. The problem of assigning semantic types to columns of tabular data is closely related to the objective of feature enhancement. For categorical column values, most approaches annotate them by exact matching to KB instances, classes, and properties. Systems such as ColNet [1] and Sherlock [3] exploit and extract contextual semantics from the tabular data using neural networks. Compared to textual data, numerical data is much harder to annotate. Most previous works develop a background KB from sources like Wikidata and then apply approaches such as transformation [11] and k-nearest neighbors search [10] to collect candidate semantic labels. These approaches are complimentary to the ideas described in this paper, and can enhance our pipeline in the data integration step.

III. System Description

Figure 1 illustrates an overview of our system. The main highlight of KAFE is that it provides a simplified access to the structured knowledge present in various knowledge graphs, as well as web sources along with an ability for the end-user to plug in their proprietary data. KAFE consists of state-of-the-art techniques to process the input data set, generate insightful visualizations and automatically identify the minimal set of features that provide maximum gain to model performance. KAFE provides transparency to inspect the internals of various algorithms. It consists of the following key components:

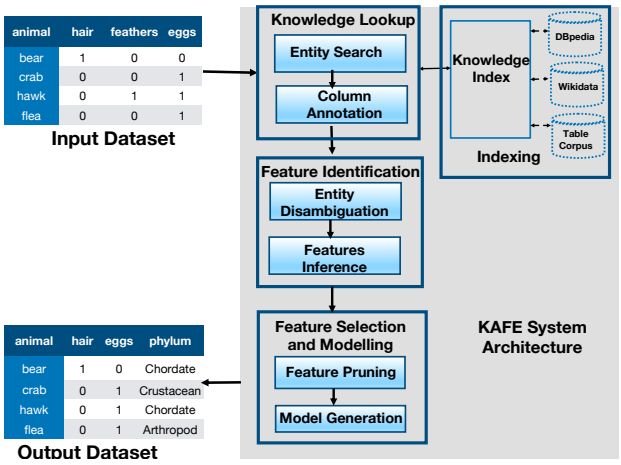


Fig. 1. KAFE System Architecture

- **Knowledge lookup**: This component iterates over different cells of the input and queries the index built on top of DBpedia, Wikidata and a collection of around 25 million web tables. The lookup component tries to identify an exact match initially and if no matches are found, it queries the index for fuzzy matches. KAFE leverages these hits to identify the semantic type of the column by voting, which is helpful for the broader understanding of the column and is useful for cell-level disambiguation in the next phase.
- **Entity Disambiguation**: The knowledge lookup phase generates different candidate hits for each cell in the table. In some cases, we find multiple hits. For example, the animal 'Duck' maps to three different entities (i) An animal (http://dbpedia.org/page/Duck), (ii) A City in North Carolina (https://www.wikidata.org/wiki/Q2724064) and (iii) A Cartoon character (http://dbpedia.org/page/Donald_Duck). KAFE disambiguates such scenarios by leveraging the semantic type of the column to construct a precise mapping between the various values in our data set to the closest knowledge graph entities and relationships. After identifying this mapping, KAFE explores the attributes of the mapped entities in knowledge graphs (or tables) to identify different candidate features. We observed that a number of DBpedia properties have multiple attribute values (For example, the animal 'duck' has more than four values for property 'type': Animal, Bird, Eukaryote, Species, etc. http://dbpedia.org/page/Duck).
- **Feature Selection and Modeling**: Once a list of candidate features is identified, KAFE performs eliminates features with few distinct values, low information gain and low correlation with the target class. The subset of features are then used to train a model. The features used to train are further pruned based on their importance values. KAFE provides the flexibility to employ other techniques to identify a small set of useful features and train a predictive model.
- **Indexing**: Given that the scale of the data available from various knowledge sources is in the millions, providing a quick access to this information was one of the key challenges for

```
In [1]: import pandas as pd
        from kbfeat.kg_helper import kg_helper

In [2]: df = pd.read_csv("datasets/zoo.data")
        df.columns

Out[2]: Index(['Animal', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '1
        1', '12',
               '13', '14', '15', '16', 'Class'],
              dtype='object')

In [3]: df.head()

Out[3]:
```

	Animal	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Class
0	aardvark	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	1
1	antelope	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	1
2	bass	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0	4
3	bear	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	1
4	boar	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	1	1

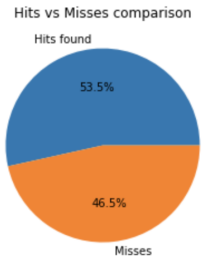
```
In [4]: sorted(df["Class"].unique())

Out[4]: [1, 2, 3, 4, 5, 6, 7]
```

(a) Data set statistics

```
In [5]: kg_obj=kg_helper(df)

        kg_obj.identify_hits('Animal')
        #Identify hits for the column 'Animal'
```



Some example hits on DBPedia:

Value	Url
Bear	http://dbpedia.org/page/Bear
Catfish	http://dbpedia.org/page/Catfish
Cheetah	http://dbpedia.org/page/Cheetah
Chicken	http://dbpedia.org/page/Chicken
Clam	http://dbpedia.org/page/Clam

(b) Data-DBpedia Mapping

Fig. 2. The Python front-end helps to easily load a data set, explore the different features and pre-process. The library is easy to invoke and it quickly shows statistics of the number of matches found between the cells of a column with the external knowledge graph. This snapshot shows the identified hits for Zoo data set (https://archive.ics.uci.edu/ml/datasets/Zoo).

our problem. We constructed an index on top of DBpedia, Wikidata data set and a dump of structured data (present in form of tables collected from various sources). In total we consider a collection of around 15 million entities. This index helps in quick lookup of an entity and examine its attributes. Given the heterogeneity of information present in various tables, our index constructs a novel homogeneous representation based on the semantic type of the information present in those data sources. Our index leverages this homogeneous representation to shortlist columns that can be joined, to speed-up the knowledge lookup and feature exploration phase.

IV. Demonstration Plan

We plan to demonstrate KAFE through an interactive Jupyter notebook[3] that will engage the audience to highlight various functions to (i) analyze the input data set, (ii) visualize and modify the mapping of entities to external knowledge, feature generation and selection, (iii) identify the smallest set of features that achieve the best performance. We will provide five pre-loaded classification and regression data sets for use. The user will go through the following different components, with an option to customize or alter the workflow as desired:

1) **Data analysis**: We provide the user, a complete handle on the data set by providing functions and visualization tools to analyze the different features. Our Python notebook allows the user to invoke any of the built-in functions in combination with our tools. The analysis of correlation and feature importance between the different features helps identify ineffective features and streamline the search for relevant domain knowledge

[3]Video of our system: https://youtu.be/7HbHeNYRh_c

```
In [6]: print ("New columns that can be added are:",kg_obj.new_feat_names)
        kg_obj.get_candidate_features()
```

```
New columns that can be added are: ['owl#differentFrom', 'soundRecording_
label', 'genus', 'synonym', 'kingdom', 'class_label', 'owl#differentFrom_
label', 'activeYearsStartYear', '22-rdf-syntax-ns#type', 'family', 'genus
_label', 'family_label', 'binomialAuthority_label', '22-rdf-syntax-ns#typ
e_label', 'order', 'birthDate', 'phylum', 'background', 'rdf-schema#seeAl
so_label', 'associatedBand', 'phylum_label', 'class', 'conservationStatu
s', 'conservationStatusSystem', 'species_label', 'binomialAuthority', 'rd
f-schema#seeAlso', 'species', 'kingdom_label', 'associatedMusicalArtist_l
abel', 'associatedBand_label', 'order_label', 'associatedMusicalArtist',
'soundRecording']
```

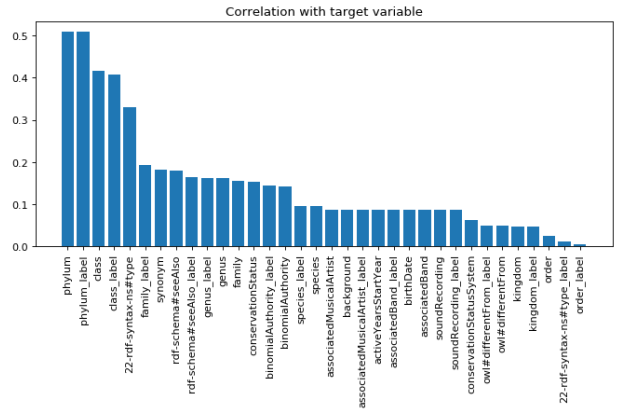
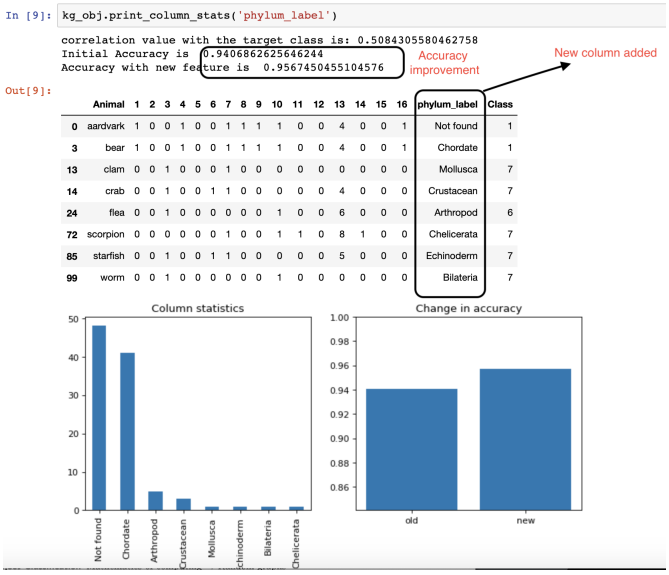


Fig. 3. The different features that can be added to the Zoo data set and their correlation with the target variable.

that can help boost the classification performance. Figure 2 displays KAFE's UI to help facilitate data loading, external knowledge mapping and analysis.

2) **Knowledge Index lookup**: KAFE provides a one-stop solution to all the data sources including DBpedia, WikiData and a collection of more than 20 million web tables. It provides



	Animal	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	phylum_label	Class
0	aardvark	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	Not found	1
3	bear	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	Chordate	1
13	clam	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	Mollusca	7
14	crab	0	0	1	0	0	1	1	0	0	0	0	0	4	0	0	0	Crustacean	7
24	flea	0	0	1	0	0	0	0	0	0	1	0	0	6	0	0	0	Arthropod	6
72	scorpion	0	0	0	0	0	0	1	0	0	1	1	0	8	1	0	0	Chelicerata	7
85	starfish	0	0	1	0	0	1	1	0	0	0	0	0	5	0	0	0	Echinoderm	7
99	worm	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	Bilateria	7

Fig. 4. A summary of the 'phylum_label' feature identified, effect on classification accuracy and the correlation with the target class.

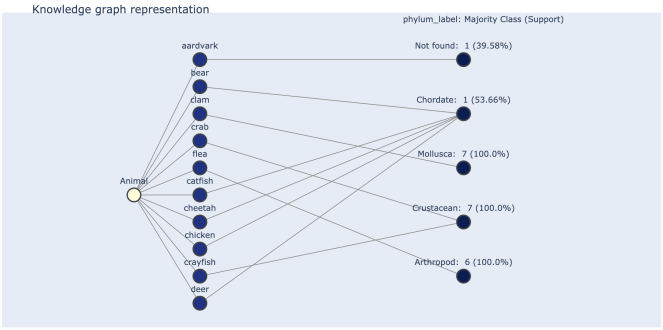


Fig. 5. The knowledge graph zoomed-in for 'Phylum' property of the identified hits along with a summary of the target class being captured by each value.

different functions to query a particular entity and explore the different hits identified. Figure 3 shows the utility of the different features to predict the target class. Figure 4 shows the deeper analysis of the particular feature, the column statistics, and classification improvements. Figure 5 shows a small subset of the knowledge graph, zoomed into the property of interest along with statistics about the target variable that is best captured by this property.

3) **Inference**: This module of KAFE demonstrates the inferred feature and its statistics e.g., correlation with the target variable, number of hits identified and change in accuracy if this feature is considered in the training phase. The different statistics and interactive visualizations generated by KAFE help the end user to quickly understand new candidate features, their summary and usefulness. KAFE exposes functionality for the user to compare the benefit of one feature on top of another by generating a correlation heat map along with change in accuracy. Figure 6 shows the correlation values between every pair of new features identified.

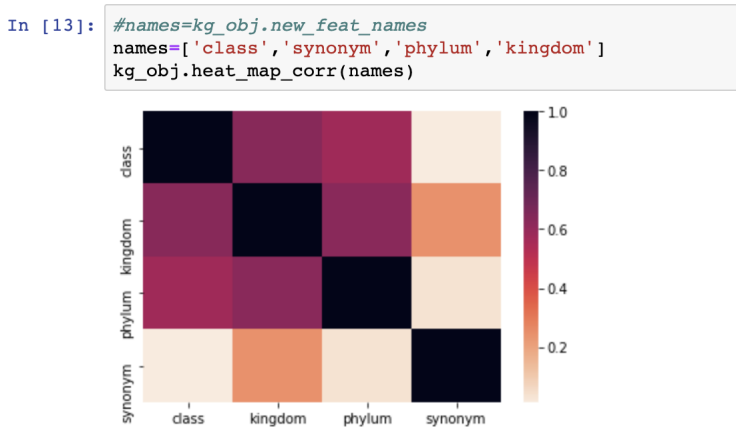


Fig. 6. Correlation heat map between the new features identified. Highly correlated features are expected to contribute towards similar examples.

4) **Feature Selection and Model Building**: This visualization component displays the accuracy improvement on considering new features identified from external knowledge sources along with feature importance. Due to space constraints, we do not show a screen shot of this aspect of our demonstration.

Summary: KAFE, a novel system, provides an insightful journey through the feature identification and inference using external knowledge sources. We expect that the broad appeal of feature enhancement and simplistic but interactive user interface of our system will help to engage users. One of the key takeaways of our demonstration would be that external knowledge sources have abundant information that can be helpful to enrich data sets for predictive modeling and KAFE provides the pathway to easily access this information at ease.

REFERENCES

[1] J Chen, E Jiménez-Ruiz, I Horrocks, and C Sutton. Colnet: Embedding the semantics of web tables for column type prediction. In *AAAI*, 2019.

[2] L Friedman and S Markovitch. Recursive feature generation for knowledge-based learning. *arXiv preprint arXiv:1802.00050*, 2018.

[3] M Hulsebos, K Hu, M Bakker, E Zgraggen, A Satyanarayan, T Kraska, Ç Demiralp, and C Hidalgo. Sherlock: A deep learning approach to semantic data type detection. *KDD*, 2019.

[4] James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *IEEE DSAA*, 2015.

[5] G Katz, E Shin, and D Song. Explorekit: Automatic feature generation and selection. In *IEEE ICDM*, 2016.

[6] U Khurana. Transformation-based feature engineering in supervised learning: Strategies toward automation. In *Feature Engineering for Machine Learning and Data Analytics*. 2018.

[7] U Khurana, H Samulowitz, and D Turaga. Feature engineering for predictive modeling using reinforcement learning. *AAAI*, 2018.

[8] U Khurana, D Turaga, H Samulowitz, and S Parthasarathy. Cognito: Automated feature engineering for supervised learning. In *IEEE ICDM*, pages 1304–1307, 2016.

[9] Fatemeh Nargesian, Horst Samulowitz, Udayan Khurana, Elias B. Khalil, and Deepak Turaga. Learning feature engineering for classification. In *IJCAI*, 2017.

[10] S Neumaier, J Umbrich, JX Parreira, and A Polleres. Multi-level semantic labelling of numerical values. In *ICWS*, 2016.

[11] P Nguyen and H Takeda. Semantic labeling for quantitative data using wikidata. 2018.

[12] H Paulheim and J Fümkranz. Unsupervised generation of data mining features from linked open data. *WIMS*, 2012.