EDITOR: Amit Sheth, amit@sc.edu

DEPARTMENT: KNOWLEDGE GRAPH

# Knowledge Graph Empowered Machine Learning Pipelines for Improved Efficiency, Reusability, and Explainability

Revathy Venkataramanan, *AI Institute, University of South Carolina, Columbia, SC, 29208, USA*

Aalap Tripathy and Martin Foltin, *Hewlett Packard Enterprise Labs, San Jose, CA, 95035, USA*

Hong Yung Yip, *University of South Carolina, Columbia, SC, 29208, USA*

Annmary Justine, *Hewlett Packard Enterprise Labs, San Jose, CA, 95035, USA*

Amit Sheth, *University of South Carolina, Columbia, SC, 29208, USA*

***Artificial intelligence (AI) pipelines are complex, heavily parameterized, and expensive to execute in terms of time and computational resources. Consequently, it is onerous to run experiments with all possible parameter combinations to achieve an optimal solution. However, these AI experiments can be optimized by recommending relevant parameters to commence the experiments, reducing search space significantly, which can be fine tuned further. The relevant parameters can be identified by observing the metadata of pipelines executed in the past, and the relevant pipeline with relevant parameters can be recommended to the user. Currently, there are various metadata frameworks that automatically record the metadata of AI pipelines. Developing a recommendation system requires understanding pipeline metadata components and their interactions. There is a need to represent the metadata generated by these AI pipelines that capture the relationship among these pipeline entities. This article presents a knowledge-infused recommender that utilizes prior knowledge and metadata of already executed pipelines represented using the proposed metadata schema to recommend a relevant pipeline per user queries. Unlike black-box models, the use of knowledge graphs makes recommendations explainable, improving transparency and trustworthiness for the users.***

Artificial intelligence (AI) pipelines are being extensively used in various research areas, such as autonomous driving,[1] DNA sequencing,[2] healthcare,[3] climate change,[4] and weather forecasting.[5] Various industries are also using them for real-time facial recognition,[6] revenue forecasting,[7] anomaly detection,[8] and other tasks. The use and prevalence of AI models are expected to increase in the upcoming decades (http://bitly.ws/xrpk). The major challenge with AI pipelines is identifying the right combination of pipeline parameters, such as data preprocessing techniques, models, and hyperparameters, to execute a given task. Illustrating with an example from Figure 1, a simple task of image classification may require the following choices to be made at the minimum: 1) choosing a model from a list of image recognition models; 2) identifying the optimal combination of data augmentations; 3) an optimizer with its own set of parameters; and 4) appropriate learning rate and its scheduling rate. In addition, there are generalization and regularization techniques that increase the search space. It is expensive in terms of time and computational resources to run all possible experiments to

1089-7801 © 2023 IEEE
Digital Object Identifier 10.1109/MIC.2022.3228087
Date of current version 3 February 2023.

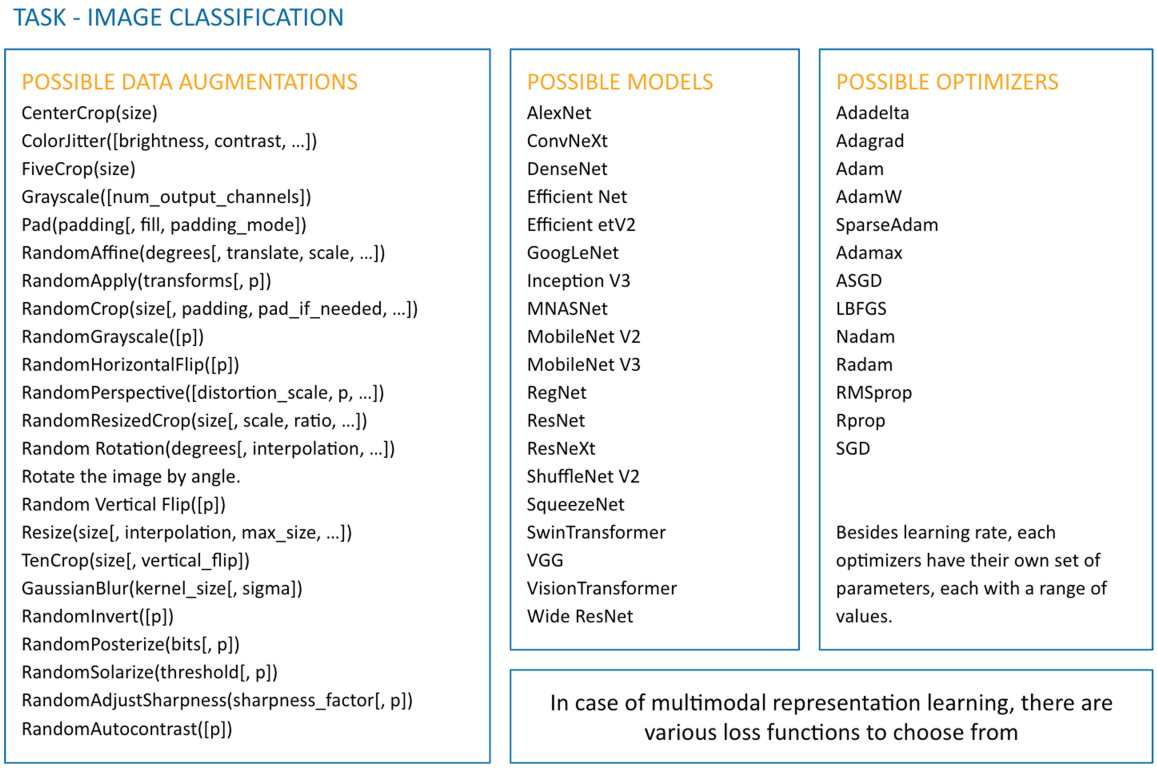


**FIGURE 1.** Number of pipeline parameters and their possible values to execute an image classification task.

achieve an optimal solution. For this open-ended problem, an optimal solution can be very subjective. In some cases, achieving the highest accuracy is the expected optimal solution. In some cases, an efficient tradeoff between accuracy and inference time can be considered an optimal solution. The machine and deep learning models are predicted to be prevalent in the upcoming years.[a] Hence, it is essential to design and implement a system that reduces the search space by recommending optimal or potential sets of pipeline parameters to commence the experiments.

The metadata collected from already executed AI pipelines can aid in determining and recommending a pipeline with relevant parameters to start the experiments and fine tune it to achieve the desired solution for a given task. Common metadata framework (CMF) is a pipeline-centric metadata framework that can collect metadata at various granular levels.[9] It automatically records the metadata of the AI pipelines, such as datasets, dataset transformations, models, hyperparameters, and the interactions among these artifacts. Using CMF, the metadata of AI pipelines can be recorded during their execution.

However, determining and recommending relevant pipeline parameters requires understanding the concepts of pipeline components, their properties, and the relationship among components. For example, suppose the user queries, *"List all the food image classification pipelines whose accuracy is more than 90%."* In that case, it requires an understanding of the following: 1) the task names can be image classification or food image classification; 2) the dataset must be of images, and the type of class labels must be food; 3) the accuracy is of type evaluation metric which should be above 90%. There is a need to represent the pipeline metadata in a schema that captures the relationships among the pipeline entities and the properties of those entities to understand such queries.

To address the abovementioned challenges, we propose a new metadata schema, built on ML-Schema,[10] to represent and store the metadata recorded from AI pipelines. Further, we propose and demonstrate a knowledge-infused recommender with a use case that utilizes the proposed metadata schema and prior knowledge to understand, identify, and recommend relevant pipelines

[a][Online]. Available: http://bitly.ws/xrpk

per the user-requested queries. While CMF can record the metadata automatically, the metadata of already executed pipelines are available from open sources, such as papers-with-code and OpenML. Similarly, open sources, such as Kaggle and HuggingFace, provide properties of pipeline entities, which comprises of both semantic and statistical features. Some examples of properties include the modality of the dataset (text, image), class of models (CNN, GRU), type of class labels of the dataset (food, vehicles, birds, or animals), number of classes in the dataset, or categories of the task (segmentation, classification). These properties constitute prior knowledge about the instances of pipeline entities, which are essential to identify relevant pipelines with similar datasets, tasks, and models to that of user-requested queries or specifications. Unlike black-box models, as the features required to identify a relevant pipeline are engineered using knowledge graphs, the recommendations made by the recommender are explainable.[11]

*Could a system recommend a relevant AI pipeline that will either suggest optimal parameters to train the model or relevant parameters to reduce the search space for the user to start their experiments and fine tune further?*

Existing neural architecture search-based approaches focus on training a deep learning model to generate and recommend deep learning architectures.[11] Several works[12] utilize the variations of Bayesian optimization to search the hyperparameter space. By recommending pipelines with past execution history, our proposed recommender differs from existing works by having the ability to reproduce the pipeline execution under a given parameter setting.

## COMPONENTS OF KNOWLEDGE INFUSED RECOMMENDER

The proposed AI pipeline recommender consists of three major components to understand the concepts and their relationships in the metadata to comprehend complex queries. It consists of: 1) a knowledge graph generated by populating the metadata schema with executed AI pipeline metadata; 2) knowledge enabled metadata collector, which extracts pipeline metadata from open sources, such as papers-with-code and OpenML; 3) a knowledge graph that consists instances of metadata components connected based on similarities of semantic and statistical properties. For example, suppose the user intends to query, *"List all the food image classification pipelines whose accuracy is more than 90%."* In that case, it requires an understanding that pipelines with the task name "image classification" and the task name "food image classification" can be considered as the dataset is about food to filter further based on accuracy. Suppose a user queries *"List all dataset and Convolutional Neural Networks (CNN) combinations that achieved above 70% accuracy in audio classification."* While CNNs are commonly used for image classification, 1-D CNNs are used for time series regressions. Such queries require an understanding that the dataset should be about audio, and CNNs are a class of models that belong to the concept "model."

### Component-1: Knowledge Graphs From Metadata Schema

In general, metadata has proven useful for personalized recommendations in various domains.[13,14] Currently, the metadata of AI pipelines is commonly used to improve the model-building process, experiment tracking, and manage model workflows.[b] On the other hand, the metadata of AI pipelines can be utilized to recommend a relevant pipeline to give a head start on experiments for a given task requested by the user. CMF, one of the metadata logging frameworks, can automatically log metadata of the AI pipelines, such as datasets, data transformations, models, hyperparameters, and interactions among these artifacts. In an overview, the components of pipeline metadata recorded by CMF are as follows: 1) task name; 2) datasets used; 3) model and its architecture; 4) hyperparameters; 5) evaluation metrics; and 6) results. To capture the relationships of these entities (components) in the pipeline and their properties, we extend the ML Schema proposed by Publio et al.,[10] as shown in Figure 2.

The schema introduced by ML-Schema was built based on OpenML pipelines, which are based on classical machine learning algorithm based. We propose an extended AI pipeline metadata schema adaptable to machine and deep learning algorithms. The proposed schema is designed to be pipeline centric rather than model centric. Further, we extend the schema to include specific statistical and semantic properties of pipeline entities that can be either populated by the user or computed from various sources based on the entity names logged by the user. The properties are essential as identifying relevant pipelines requires both statistical and semantical properties of pipeline entities. We include additional entities to our metadata schema, such as execution time, framework, and deployment parameters.

[b][Online]. Available: http://bitly.ws/xrpp

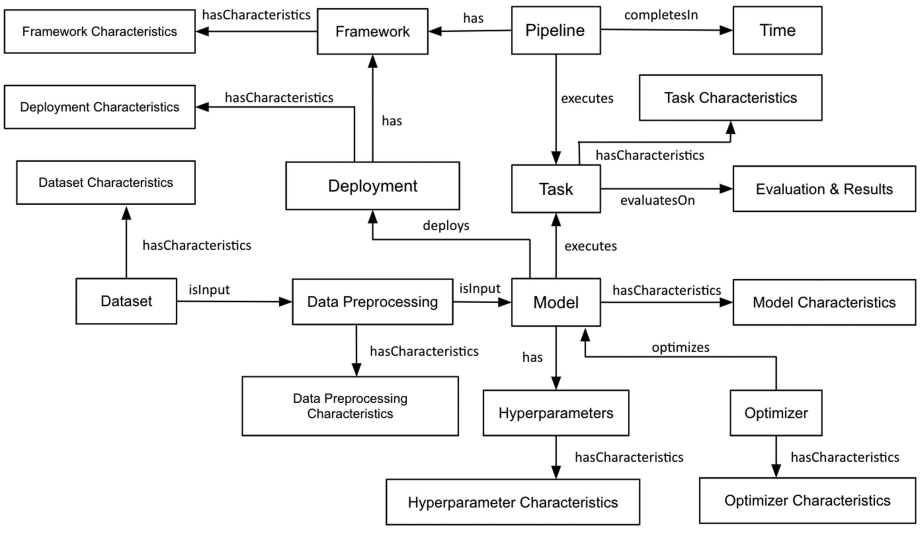


**FIGURE 2.** Proposed metadata schema for AI pipelines built on ML-Schema.[10] It includes specific statistical and semantic characteristics for all schema entities. Further, it is extended to include entities, such as deployment, data preprocessing, and framework, along with its characteristics.

## Component-2: Knowledge-enabled Metadata Collection

The metadata recorded using CMF proved to be significant in identifying the model and hyperparameter settings for various experiments we conducted. It is well known that generating several instances of AI pipeline metadata with various machine and deep learning models demands significant resources in terms of time and computation. At the same time, open sources, such as papers-with-code[c] and OpenML,[d] have metadata of already executed pipelines. Leveraging that, the metadata offered by these open sources was collected. Notably, papers-with-code is up-to-date with state-of-the-art models and their experiments reported by several published papers. OpenML consists of pipelines executed and their metadata recorded by its users. Most of the pipeline metadata recorded by OpenML consists of machine learning models compared to deep learning models.

Both the open sources expose the metadata of the pipelines through their application programming interface (API). Papers-with-code consists of 1 million pipelines in the form of published papers. The unique entities include 3800 tasks, 11,800 datasets, 7875 evaluations, and result tables. The details of the pipeline's hyperparameters can be accessed through their git repository and the published paper manuscript, which is provided by papers-with-code. OpenML consists of 10 million pipelines for machine learning models recorded by various users.

The nomenclature and data format provided by both the open sources were different, and it posed a challenge in integrating the data sources. First, the data from both open sources are collected and stored. Then the concepts from papers-with-code and OpenML are mapped to the concepts in the metadata schema created for CMF (see Figure 2). Through further analysis, we found that of the one million papers, only around 18,000 papers have complete data available through the API. Similarly, for OpenML, the hyperparameters are not logged by some users. Further, most OpenML pipelines are machine learning models, such as support vector machines and logistic regression models. This poses a problem of data incompleteness and lack of diversity. Hence, we propose to build a parser that extracts metadata from published papers available from papers-with-code and arxiv.

To extract the desired metadata pipeline entities from the published papers, we build taxonomy of vocabulary similar to SNOMED CT.[15] We curate a vocabulary for the following components: 1) dataset; 2) task; 3) model; 4) hyperparameters; 5) evaluations; and 6) results. We utilize the pipeline metadata recorded through CMF and pipeline metadata available through API from papers-with-code, OpenML, and HuggingFace to collect datasets, tasks, model names, evaluation metrics, and results. We use Kaggle to collect datasets and deep learning libraries, such as Pytorch, Tensorflow, and Keras, to collect and curate hyperparameter taxonomy. The extracted data will enrich the vocabulary iteratively, as shown in Figure 3.

[c][Online]. Available: https://paperswithcode.com/
[d][Online]. Available: https://www.openml.org/

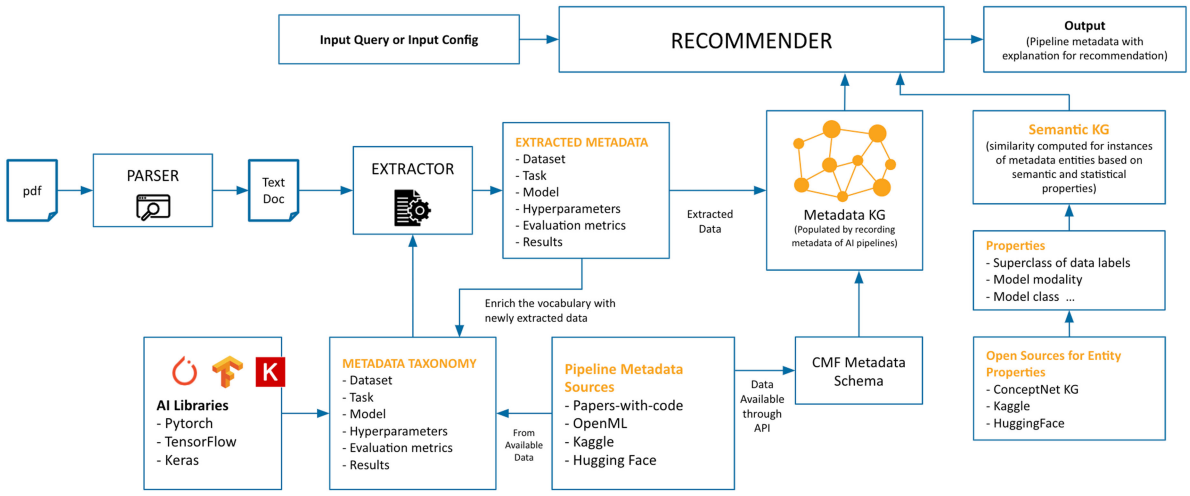


**FIGURE 3.** System architecture of the proposed knowledge-infused recommender system. The metadata KG is populated using data from open sources through the metadata schema (see the "Component-1: Knowledge Graphs From Metadata Schema" section). The semantic KG (see the "Component-3: Knowledge Graph for Semantics" section) is created by computing similarity among the instances of various entities of the pipeline metadata using their statistical and semantic properties.

## Component-3: Knowledge Graph for Semantics

We gather the properties for all the instances of each entity in the metadata pipeline to identify semantically relevant pipelines and to provide an explanation for the recommendation produced by the system. The user can log the properties of each metadata entity, such as task modality, task category, dataset modality, and so on. On the other hand, the value of such properties can be computed using NLP techniques and collected from other open sources. For example, for a given data label cat, the superclass of the data label can be identified as animal using the type of relation from ConceptNet. This is necessary to determine what the dataset represents and recommend a similar dataset as per the user's request (discussed in the "Workflow" section). Furthermore, we curated a vocabulary from available task names to identify task modality and task category for unseen task names. The open sources Kaggle and OpenML provide statistical properties of a dataset through their API. Similarly, HuggingFace, Pytorch, TensorFlow, and Keras can provide properties of off-the-shelf models, such as model class (CNN, transformer), model modality (image, text, multimodal), the number of model parameters, and the model architecture.

The proposed end-to-end system is a living pipeline that collects data from open sources at regular intervals, and dynamically updates our knowledge graphs and data stores. To the best of our knowledge, this is the first AI pipeline recommender system with various comprehensive knowledge graphs curated using data from multiple sources.

## WORKFLOW

We illustrate the proposed recommender system workflow with a sample case scenario as follows.

The proposed system can map the desired entities from the natural language query entered by the user to the entities in the proposed metadata schema. In this example, we demonstrate the ability of our recommender to identify a similar dataset if the dataset requested by the user is unavailable. Further, the system can determine appropriate task names (image classification and food image classification) based on the superclass of dataset class labels (food). The user has requested CNN to conduct the experiments. The system can identify this as a property *model class* of the entity *model* and filter the image classification pipelines that use only CNN models, such as ResNet, AlexNet, and VggNet. The proposed recommender can support queries to identify similar tasks, deployment parameters, or data preprocessing techniques. Additional information on the schema and workflow examples can be found online.[e] The knowledge graphs used for

[e][Online]. Available: https://hewlettpackard.github.io/cmf/schema

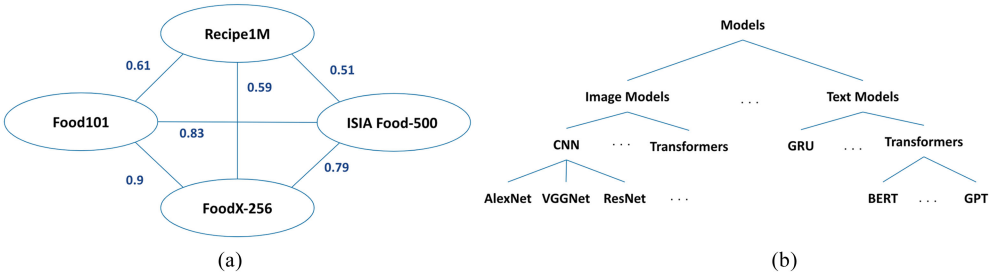


**FIGURE 4.** Illustration of a subset of knowledge graphs used for the recommendation. (a) Illustration of dataset similarity graph with weighted edges where the similarity is calculated based on both statistical and semantic features. (b) Illustration of model hierarchy tree that consists of a list of models under a given model class.

| *Query: "List all the image classification pipelines with Food-101 dataset, CNN models, and accuracy above 90%."* | |
|---|---|
| *Step 1: Converting the query using language models* | *Step 3: Sorting and filtering using knowledge graphs in Figure 4* |
| Task: Image classification | 1) Fetch all pipelines with task name "Image Classification."<br>2) Since the dataset label has only one quantity, also search for pipelines with the name "Food Image Classification."<br>3) Filter the pipelines with "Food-101" as the dataset name.<br>4) If the exact match of the dataset is not found, identify similar datasets as follows.<br>a) Identify datasets with the label only as food.<br>b) Identify datasets with data modality as image.<br>c) Identify datasets with similar statistical and semantic properties, such as number of class labels, number of images per label, image size, image color, and so on.<br>d) Based on these features, filter the pipelines further and rank them [see Figure 4(a)].<br>5) Filter the pipelines with model class name as CNN, which can be ResNet, AlexNet, or VGGNet [see Figure 4(b)].<br>6) Filter the pipeline again based on evaluation metric as accuracy and result above 90%.<br>7) Rank the pipeline based on the feature matches.<br>8) Present the recommendation to the user with the list of ranked pipelines and ranking their scores. Further, present the explanation by listing the features used to identify the relevant. |
| Dataset: Food-101 | |
| Accuracy: 90% | |
| Model: CNN | |
| *Step 2: Fetching properties using components 1 and 3* | |
| Task name: Image classification | |
| Task modality: Image | |
| Task category: Classification | |
| Dataset name: Food-101 | |
| Dataset modality: Image | |
| Dataset label list: Food | |
| Image size: 100x100 | |
| Image color: RGB | |
| Evaluation metric: Accuracy | |
| Result: 90% | |
| Model class: CNN | |

relevant pipeline recommendations capture the pipeline entities' relationship. The features used to determine relevant pipelines are explicitly engineered based on the properties of these pipeline entities. Thus, the recommender can explain similarity computation to identify relevant pipelines, and the choice of relevant pipelines can be explained to the user.

## RESEARCH CHALLENGES

*Feature engineering and explainability:* Determining relevant pipelines based on keyword match of entity name will not be sufficient. For example, using keyword matches, we cannot identify other food image classification datasets (FoodX-256) besides the user-requested dataset, say Food-101. This will significantly limit the capabilities and usage of the recommender. The semantic and statistical features of each entity in the pipeline metadata must be considered for meaningful recommendation. Curating and computing values of these properties are challenging due to the lack of structured data about pipeline metadata entities. We propose to gather information from open sources and knowledge graphs, such as Kaggle, HuggingFace,

## COMMON METADATA FRAMEWORK

It is well established that AI pipelines can benefit from recording their metadata, which can be used for experiment tracking, model tuning, and deployment. Metadata tracking frameworks in the market broadly fall under two categories. Frameworks like MLFLow[1] and Weights and Biases[2] are experiment centric, and ML Metadata[3] is developed based on a pipeline-centric approach.

Common metadata framework (CMF)[4] is built on top of ML Metadata and takes a pipeline-centric approach while incorporating features from experiment-centric frameworks. It can automatically record AI pipeline metadata from different stages in the pipeline and offers fine-grained experiment tracking like experiment-centric frameworks. Different stages in an AI pipeline can be spread across different sites, and each of these stages may have different experiment variants. It is important to track metadata for each experiment variant for reproducibility, audit trail, and traceability. This lineage metadata tracked for various artifacts in the pipeline plays a crucial role in the recommendation.

Lineage tracked using CMF provides illuminating insights into the artifacts. One such example is the qualitative and statistical information about the dataset used to train the model provides indicators to measure the trustworthiness and biasedness of the model. This enables recommendations based on the comprehensive assessment of the quality of the model based on its entire lifecycle and not just the final metrics.


### REFERENCES

1. "Machine learning flow tracking," MLFlow, 2022. [Online]. Available: https://www.mlflow.org/docs/latest/tracking.html
2. "Weights and biases," WanDB, 2022. [Online]. Available: https://wandb.ai/site/experiment-tracking
3. "Machine learning metadata," TensorFlow, 2022. [Online]. Available: https://www.tensorflow.org/tfx/guide/mlmd
4. A. Justine et al., "Self-learning data foundation for scientific AI," in *Proc. SMC*, Springer, 2022.


Pytorch, Tensorflow, and ConceptNet, to compute the values of these properties.

*Integrating metadata from multiple sources:* The pipeline metadata open sources, such as papers-with-code, OpenML, and HuggingFace, expose the metadata of AI pipelines through their API with different nomenclature and schema. For example, *models* are named as *methods* in papers-with-code and as *flows* in OpenML. Mapping the entities provided by the open sources to the concepts in metadata schema is challenging as we plan to include more open sources. We aim to build a vocabulary to map different nomenclature from multiple sources to the concepts in the metadata schema.

*Parsing and extracting metadata:* As the open sources expose only limited or incomplete pipeline metadata through their API, we propose to build a parser, as described in the "Component-3: Knowledge Graph for Semantics" section. This demands a curation of an extensive vocabulary. For example, the authors of published manuscripts might use the term *lr* or *LR* or simply *learning rate* to define the learning rate utilized in their experiments. In addition, different conferences use different templates for their papers. It is necessary and challenging to build a parser that can manage different paper templates to extract the metadata of AI pipelines.

## FUTURE WORK AND CONCLUSION

AI pipelines are hard to train as they require the selection of appropriate combination parameters from the large search space. In this work, we demonstrated recommendations of relevant AI pipelines with a use case to reduce the search space of the parameters to commence the experiments. This living framework, can dynamically update the knowledge graph and the data store as new data are generated. The recommendation is explainable as the features used to identify a relevant pipeline are explicitly engineered. Therefore, due to its transparency, the recommender system is also trustworthy to its users. In our future work, we aim to improve and prioritize the recommendations based on user feedback through metalearning approaches. We also propose to enhance the system to suggest knowledge-infusion techniques for the recommended pipeline or suggest datasets that can be used to train the models in the recommended pipeline through semisupervised and unsupervised approaches.


## ACKNOWLEDGMENTS

The proposed work is done as collaboration between the University of South Carolina and Hewlett Packard Enterprise. The authors sincerely thank Suparna Bhattacharya Sergey Serebyrakov, Tarun Kumar, Ashish Mishra, Arpit

Shah, Paolo Faraboschi, Kunal Sharma, and Emily Cheng for their iterative suggestions and feedback. The author would like to acknowledge National Science Foundation (NSF) Awards #2133842 "EAGER: Advancing Neuro-symbolic AI with Deep Knowledge- infused Learning," and #2119654 "RII Track 2 FEC: Enabling Factory to Factory (F2F) Networking for Future Manufacturing" in part for the support. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the company and funded agency.

## REFERENCES

1. E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE Access*, vol. 8, pp. 58443–58469, 2020, doi: 10.1109/ACCESS.2020.2983149.
2. B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning," *Nature Biotechnol.*, vol. 33 no. 8, pp. 831–838, 2015.
3. F. Jiang et al., "Artificial intelligence in healthcare: Past, present and future," *Stroke Vasc. Neurol.*, vol. 2 no. 4, 2017.
4. S. Fathi, R. S. Srinivasan, C. J. Kibert, R. L. Steiner, and E. Demirezen, "AI-based campus energy use prediction for assessing the effects of climate change," *Sustainability*, vol. 12 no. 8, 2020, Art. no. 3223.
5. S. Dewitte, J. P. Cornelis, R. Müller, and A. Munteanu, "Artificial intelligence revolutionises weather forecast, climate monitoring and decadal prediction," *Remote Sens.*, vol. 13 no. 16, 2021, Art. no. 3209.
6. Y. Zhong, S. Oh, and H. C. Moon, "Service transformation under industry 4.0: Investigating acceptance of facial recognition payment through an extended technology acceptance model," *Technol. Soc.*, vol. 64, 2021, Art. no. 101515.
7. R. I. Whitfield and A. H. B. Duffy, "Extended revenue forecasting within a service industry," *Int. J. Prod. Econ.*, vol. 141 no. 2, pp. 505–518, 2013.
8. L. Stojanovic, M. Dinic, N. Stojanovic, and A. Stojadinovic, "Big-data-driven anomaly detection in industry (4.0): An approach and a case study," in *Proc. IEEE Int. Conf. Big Data*, 2016, pp. 1647–1652.
9. A. Justine et al., "Self-learning Data Foundation for Scientific AI," in *Proc. Sound Music*, 2022, pp. 4–11.
10. G. C. Publio et al., "ML-schema: Exposing the semantics of machine learning with schemas and ontologies," 2018, *arXiv:1807.05351*.
11. T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20 no. 1, pp. 1997–2017, 2019.
12. H. Zhou, M. Yang, J. Wang, and W. Pan, "BayesNAS: A Bayesian approach for neural architecture search," in *Proc. Int. Conf. Mach. Learn*, PMLR, May. 2019, pp. 7603–7613.
13. A. Nechaev, V. Meltsov, and N. Zhukova, "Utilizing metadata to select a recommendation algorithm for a user or an item," in *Proc. CEUR Workshop Proc.*, 2020.
14. A. Nauerz, F. Bakalov, B. König-Ries, and M. Welsch, "Personalized recommendation of related content based on automatic metadata extraction," in *Proc. Conf. Center Adv. Stud. Collaborative Res.: Meeting Minds*, Oct. 2008, pp. 57–71.
15. K. Donnelly, "SNOMED-CT: The advanced terminology and coding system for eHealth," *Stud. Health Technol. Inform.*, vol. 121, 2006, Art. no. 279.

**REVATHY VENKATARAMANAN** is a Ph.D. student advised by Prof. Amit Sheth at AI Institute, University of South Carolina, Columbia, SC, 29208, USA. She is also a research intern at Hewlett Packard Enterprise Labs, San Jose, CA, 95035, USA. Contact her at revathy@email.sc.edu.

**AALAP TRIPATHY** is a principal research engineer at Hewlett Packard Enterprise Labs, San Jose, CA, 95035, USA, working on edge-to-exascale workflows and data foundation for AI workflows. His experience spans HPC, AI, IoT & edge computing, parallel and distributed processing & semantic computing. Contact him at aalap.tripathy@hpe.com.

**MARTIN FOLTIN** leads the AI infrastructure team at Hewlett Packard Labs, San Jose, CA, 95035, USA. His experience spans AI, silicon design, and molecular physics. Contact him at martin.foltin@hpe.com.

**HONG YUNG YIP** is a Ph.D. student advised by Prof. Amit Sheth at AI Institute, University of South Carolina, Columbia, SC, 29208, USA. Contact him at hyip@email.sc.edu.

**ANNMARY JUSTINE** is a research engineer at Hewlett Packard Enterprise Labs, San Jose, CA, 95035, USA, with more than 12 years of experience in various storage and AI products. Her current focus is on data centric and trustworthy AI. Contact her at annmary.roy@hpe.com.

**AMIT SHETH** is the founding director of the AI Institute (AIISC), NCR Chair and Professor of computer science and engineering at the University of South Carolina, Columbia, SC, 29208, USA. He is a fellow of IEEE, AAAS, AAAI, and ACM. Contact him at amit@sc.edu.