Hadoop-MTA: a system for Multi Data-center Trillion Concepts Auto-ML atop Hadoop

Keqian Li[†], Yifan Hu[†], Manisha Verma[×], Fei Tan[†], Changwei Hu[¶], Tejaswi Kasturi[†], Kevin Yen[†]
Yahoo Research[†], Amazon Inc.[×], XPeng Motors[¶], RED [†],

***Abstract*—The ever-growing computation capability distributed infrastructure brings tremendous opportunities for mining and analysis of data that was impossible otherwise. Meanwhile, the inherent computation model of distributed system also brings unique and non-trivial challenges for traditional Auto-ML, including the explosion of data dimensions, the expected absence of features, and the heterogeneity of information. This is especially the case in modern Internet enterprises, where data in the scale of trillions are stored in multiple data centers, and the discovery of subtle signals could incur significant impact in revenue and welfare. How can we best harness the large scale distributed machine learning, but without keeping engineers constantly in the loop? In this work, we present *Hadoop-MTA*, a system for Multi Data-center, Trillion Concepts, Auto-ML on top of the Hadoop distributed computation environment that leverages sparsity aware heterogeneous knowledge graph representation and dimensionality agnostic parallel learning. Through multiple large scale experiments, we find that *Hadoop-MTA* significantly outputperforms competitive state of the art distributed learning algorithms and scales well to trillion scale data-sets. Our model is rolled out to Hadoop serving infrastructure in Yahoo covering billions of unique identities and shows improvements 129.5% accuracy and 106.5 % weighted F1-score (more than 2x) on key targeting use cases.**

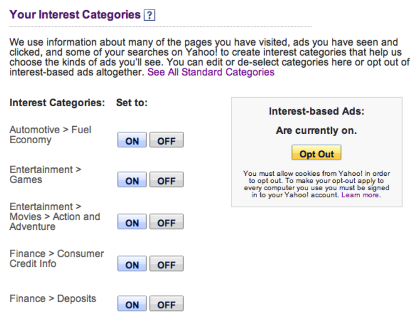
I. Introduction

Ever since the introduction of commercial distributed computation engine such as Hadoop, distributed system has transformed the industry landscape by scaling up the ingesting, analysis and serving infrastructure and supporting large scale modeling use cases such as personalization and online advertising [1]. However, the sheer scale of the computation and overheads also incur significant turnaround time and may requires engineers and scientists constantly engaging during the ML life-cycle and incur significant operating costs. The importance Auto-ML cannot be overstated in this case.

Traditionally Auto-ML paradigm [2] fails in distributed computation use cases primarily due to the following reasons:

- **Data Integration**: digital footprints such as instrumentation logs are collected and integrated from disparate domain and even different physical data centers with irregular schema relationships and modality, which presents significant challenges for models in traditional domains such as vision, language, speech, time series, spreadsheet where data are expected with regularly shapes
- **Dimension Explosion**: the unlimited horizontal scalability of multi data center distributed system leads to $O(n)$ growth of data dimensionality unforeseen in traditional scalable learning approach such as data-parallel learning and stochastic optimization
- **Expected Absence**: vast majority of the digital footprints are in the form of implicit feedback where features are expected to be *missing at random* (MAR) where learning algorithm for dense data suffers from efficiency or effectiveness drop

We present *Hadoop-MTA*, a system for Multi Data-center, Trillion Concepts AutoML on top of the Hadoop distributed computation environment. Our system builds upon previous concept mining and analysis stack [3], [4], [5], [6], [7]. Instead of optimizing specific part of the pipeline [8] and specific modality [9], [10], [11], [12], we focus on end to end solution for distributed AutoML that scales above trillions of concepts, through computation space and time preserving transformation a unified unfolded concept learning framework and present subsequent AutoML pipelines with provable performance guarantee.


Your Interest Categories ?

We use information about many of the pages you have visited, ads you have seen and clicked, and some of your searches on Yahoo! to create interest categories that help us choose the kinds of ads you'll see. You can edit or de-select categories here or opt out of interest-based ads altogether. See All Standard Categories

Interest Categories:	Set to:
Automotive > Fuel Economy	ON OFF
Entertainment > Games	ON OFF
Entertainment > Movies > Action and Adventure	ON OFF
Finance > Consumer Credit Info	ON OFF
Finance > Deposits	ON OFF

Interest-based Ads:
Are currently on.
Opt Out
You must allow cookies from Yahoo! in order to opt out. To make your opt-out apply to every computer you use you must be signed in to your Yahoo! account. Learn more.

Fig. 1: Illustration of Hadoop-MTA use cases. Personalization insights are provided along with options for opt-out

	Accuracy	AUC	Recall	Prec.	F1	Improvement from precedent
Candidate production	0.569	0.569	0.55	0.75	0.6	-
with activity feature	0.815907	0.61173	0.991181	0.821024	0.898114	+ 7.51%
plus stochastic AutoML	0.823101	0.637293	0.995811	0.825528	0.902709	+ 4.18%
plus data integration	0.8131	0.7489	1	0.8131	0.8969	+ 17.51%
plus compliant label engineering *	0.8272	0.8419	0.9998	0.8271	0.9053	+ 12.42%

TABLE I: Comparison of *Hadoop-MTA* with candidate production system on key rare-class predictive segment task. Method marked with * are evaluated on different non-identical evaluation set.

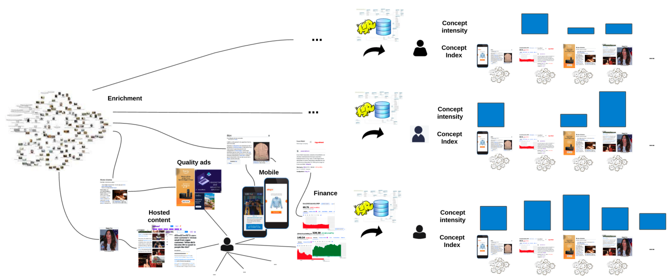


Fig. 2: Illustration of *concept unfolding* for transforming heterogeneous data into sparse sequence of concepts

II. Concept unfolding

As illustrated in Figure 2, different objects of interests such as raw events of user activity are logged into production, which may go through intermediate enrichment via automated or manual labeling

	Accuracy	**Recall**	**Precision**	**Weighted F1**	**Cohen Kappa**
Previous Production	0.1954	0.2327	0.2834	0.2148	0.0764
Hadoop-MTA	0.4485	0.3501	*0.4434*	0.4437	0.3022
Lift	+129.5%	+50.5%	+56.4%	+106.5%	+295.5%

TABLE II: Comparison of *Hadoop-MTA* with the previous production system on accuracy based targeting use case

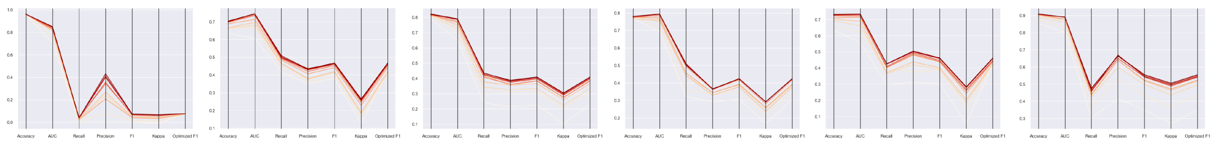


Fig. 3: Evaluation of *Hadoop-MTA* black-box in-the-core AutoML following exponential search schedule

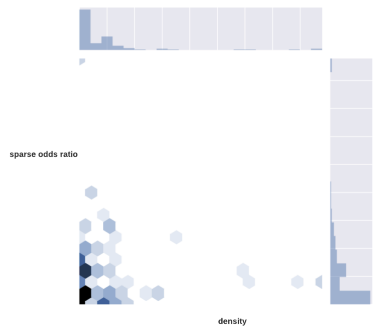


Fig. 4: Second order join distribution of concepts

and linked to other external entities and further interconnected with others objects such as events, users and entities. In this case, data are assumed to be of arbitrary schema and interlinked in arbitrary ways of their choice, and physically resides in arbitrary types of data storage engines in Hadoop. We start by exploring different options to represent the data.

Formally, assume the objects are organized as entities $\mathcal{S}$. According to the type of entity $\mathbb{T}(s) \in \mathcal{T}$, the meta data of entity $s.feature$ may be a scalar, ordered list, dictionary or recursive composition of them of fixed schema $\Psi_{\mathbb{T}(s)}$. There is also a set of link $l \in \mathcal{L}$ that connects two or more entities whose metadata $l.feature$ may also be a scalar, ordered list, dictionary or recursive composition of them of fixed schema $\Psi_{\mathbb{T}(l)}$ depending on its type $\mathbb{T}((l)$. In this work, we focus on learning for entity level predictions:

Definition 1 (Learning in heterogeneous data): Assuming the label function of interest $y : \mathcal{S} \to \mathcal{Y}$ mapping entity to one label in $\mathcal{Y}$, the goal is to learn a hypothesis $h_{\mathcal{S},\mathcal{L},\mathbb{T}} \in \mathcal{H} \subseteq \mathcal{S} \to \mathcal{Y}$ that minimizes its expected risk from the label by a given standard $\mathbf{L}$ under a probability measure of the entity $p : \mathcal{S} \to [0, 1]$

$$\underset{h_{\mathcal{S},\mathcal{L},\mathbb{T}}}{\text{minimize}}\ R(h_{\mathcal{S},\mathcal{L},\mathbb{T}}) = \mathbb{E}[\mathbf{L}(h_{\mathcal{S},\mathcal{L},\mathbb{T}}(s), y(s))]$$
$$= \int_{\mathcal{S}} \mathbf{L}(h_{\mathcal{S},\mathcal{L},\mathbb{T}}(s), y(s))p(s)ds$$

For traditional relational data such as database, spreadsheets, CSV [8] one can formalize a similar learning task as below:

Definition 2 (Learning in relational database): assume the objects are organized as entities $\mathcal{S}$ and links $l \in \mathcal{L}$ similar as above, except the meta data of entity $s.feature, l.feature$ is ordered list of scalars of fixed schema $\Psi_{\mathbb{T}(s)}$, $\Psi_{\mathbb{T}(l)}$, respectively. The goal is to learning a function $h_{\mathcal{S},\mathcal{L},\mathbb{T}} \in \mathcal{H}_R$ minimize the risk $R(h_{\mathcal{S},\mathcal{L},\mathbb{T}})$ in the same form as above.

Furthermore, we also study the important modalities from heterogeneous graph and graph neural network literature [13], as well as first-order logic knowledge-base.

Definition 3 (Heterogeneous graph learning): A *heterogeneous network* $G = \{\mathcal{S}, \mathcal{L}, X, R, \mathbb{T}\}$ is a network with multiple types of nodes and links. Particularly, within H, each node $v_i \in \mathcal{S}$ is associated with a node type $\mathbb{T}(v_i)$, and each link $e_{ij} \in \mathcal{L}$ is associated with a link type $\mathbb{T}(e_{ij})$. Each node v_i of type $o = \phi(v_i)$ is also potentially associated with attribute X_i^o, while each link e_{ij} of type $o = \psi(e_{ij})$ with attribute R_{ij}^o. The goal of Heterogeneous graph learning is to learn a function $h_G \in \mathcal{H}_G \subseteq \mathcal{S} \to \mathcal{Y}$ that minimize the loss R defined same as above.

Definition 4 (First Order logic inference): Given a set of variables $\mathcal{S}$, true valued-expression $\mathcal{L}$ consisting of variables and their recursive composition via first order logic connectives, quantifiers and functions, the goal of first order logic knowledge graph inference is to find the truth value for other expressions $\mathcal{L}'$ also consisting of variables and their recursive composition via first order logic connectives, quantifiers and functions.

We present the *Unfolded concept learning* framework as a unified framework that pave the way for advanced mining and Auto-ML techniques, and a scalable distributed algorithm to implement this. As shown in Figure 2, each particular multi-hop connection, for example one conforming to meta-path [13] or a SPARQL query, is serialized as an element c in concept vocabulary, $c \in \mathcal{C}$. where a feature vector $\vec{c_s}$ with a bijective mapping between its index and $\mathcal{C}$ are stored for each entity $s \in \mathcal{C}$. If we again simplify the notation and set $s.feature := \vec{c_s}$ for all $s \in \mathcal{S}$, we have

Definition 5 (Unfolded concept learning): For each entity $s \in \mathcal{S}$, the goal is to learn a model $h \in \mathcal{H}_C \subseteq \mathbb{R}^{\mathcal{C}} \to \mathcal{Y}$, that minimize the risk defined in Definition 1.

By first emitting individual concepts and perform *reduce* to create a vector for each entity, we have the following efficiency result

Theorem 1 (Existence of $O(d)$ time and space concept unfolding in distributed computation): Assuming the distributed primitive of map, reduce, there exists algorithm for producing unfolded concepts for every entity in d time where d is the degree of the data, defined as maximum number of non-zero concepts associated with an entity $s \in \mathcal{S}$.

This unfolded concept learning framework allows efficient integration of data while leveraging the absence of feature to support sparsity preservation computation through the AutoML pipeline. Furthermore, it efficiently represent all the above problems where architecture such as GNN [13] can be included as a special case in the unfolded concept learning solution space.

Theorem 2: The above learning problems (Definition 1-4) accepts polynomial reduction to the concept learning as in Definition 5. Specifically, there exists linear time reduction from learning in heterogeneous data, relational database, and heterogeneous graph learning (Definition 1-3) .

III. Concept Validation

The quality of mined concepts and data-set are of primary importance to downstream tasks. For this purpose we follow previous literature on concepts mining [3], [9] and specifically propose the following criteria for Auto-ML:

- **Concept Reliability**: are concept's occurrence too rare to be found useful?
- **Concept Informativeness**: are concept's informative to the meaning of an instance?
- **Concept Relevance**: are concept particularly relevant to the task at hand?
- **Instance Reliability**: are the instance providing reliable and meaningful information for the model to predict?

We leverage the following metrics for the above criteria:

- **Concept Support** is defined as the absolute number of occurrence of a concept and are used as a high-pass filter.
- **Concept Density** is defined as the likelihood of occurrence of a concept and are used as a low-pass filter.
- **Instance Support** is defined as to the number of concepts of an instance and are used as a high-pass filter. . Unreliable instances are likely associated with low instance support.
- **Sparse Odds ratio** is defined as the relative likelihood of occurrence of a concept in the sparse representation of instance excluding zero values, conditioned on different value of label. Concept with high value will be highly relevant.

Although the above statistics are simple to collect for in-the-core computation, directly applying distributed algorithm, such as distributed Pearson score computation [14] will be disastrous precisely due to *Dimension Explosion*. To address this, a efficient frequent concept mining algorithm was implemented that parallels by the dimension of features leveraging the sparsity of concept possible representation. It outputs frequent concepts by emitting local frequency of all concepts tuples up to order of 2 and calculates the statistics of the joint distribution of all concepts at the same time, including the support, odds ratio and prevalence defined above. Details will be found on extended version of the paper.

Theorem 3 (Existence of $O(1)$ second order join distribution statistics in distributed computation): for distributed , assuming the cardinality is finite, then only need to enumerate Assuming the distributed primitive of map, reduce and combine, there exists algorithm for producing second order join distribution statistics for every concepts in $O(1)$ excluding the $O(d)$ emit operation in map, where d is the degree of the data defined in Theorem 1.

Application to label engineering The second order join distribution statistics can also be applied to label engineering, to efficiently select agreement among multiple agency and data sources in the distributed computing environment. AS a result, we can automate the label engineering effort and impact downstream metrics significantly (See Table I).

IV. Concept learning

Given the concept vector $\vec{c_s}$ for each $s \in \mathcal{S}$, it is relatively straight-forward to adapt parallel learning algorithms implemented in Hadoop or Spark or stochastic learning algorithms with Hadoop I/O. Furthermore, we implemented a black-box in-the-core AutoML algorithm that runs simultaneous reduce operation, each aggregating a_i of mapper results from a given series $(a_1, a_2, a_3, \ldots)$ and runs in-the-core AutoML algorithm to serve as candidate model for cross validation and ensembling. By following an exponentially increasing search schedule, such that $(a_1, a_2, a_3, \ldots) := (a_1, C\cdot a_1, C^2\cdot a_1, \ldots)$, we have the following performance and efficiency guarantee for the black-box in-the-core AutoML:

Theorem 4 (Existence of on black-box in-the-core AutoML): For any hypothesis h drawn from hypothesis space $\mathcal{H}$ with training complexity $\Omega(n)$ where n is the training set size , the difference between training error $\epsilon_n(h)$ and generalization error $\hat{\epsilon}(h)$ is bounded for any fixed probability

$$|\epsilon_n(h) - \hat{\epsilon}(h)| = \mathcal{O}\left(\sqrt{\frac{\gamma}{n(\gamma-1)}\log(\frac{n(\gamma-1)}{\gamma})}\right)$$

where γ is the ratio between the actual computation flops compared to the maximum computation flops that fits into in-the-core computation.

The final Auto ML system is a hybrid that selects the best among all the above methods.

V. Evaluation

The Hadoop-MTA is rolled out to production targeting use cases to cover billions of user traffic per month. Evaluation in key AUC-based classification task (Table I) and accuracy-based bucket prediction task (Table II) shows significant performance gains over production systems. Individual ablation test over components for second order joint distribution of concepts where the sparse odds ratio (with 0 value at bottom and 9 at the top) and the density (with 0 value at left to 1 at the right) of concepts are jointly plotted (Figure 4) for the AUC-based classification task; for black-box in-the-core AutoML with exponential search, where bigger searches are encoded with darker color (Figure 3) for six key AUC-based classification task; and for individual component improvements where a URL based candidate production system is improved by replacing with feature with tracked activity, stochastic AutoML that incrementally learns from batch of data from distributed storage, data integration across multiple data centers to join together URL and activity data sources, and compliant label engineering that applies label engineering (Section III) with only compliant data sources for training and additional data sources for evaluation.

References

[1] B. Balusamy, S. Kadry, A. H. Gandomi *et al.*, "Driving big data with hadoop tools and technologies," 2021.

[2] Y. Li et.al., "Automl: From methodology to application," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 4853–4856.

[3] K. Li, "Mining and analyzing technical knowledge based on concepts," Ph.D. dissertation, University of California Santa Barbara, 2019.

[4] H. Zha, W. Chen, K. Li, and X. Yan, "Mining algorithm roadmap in scientific publications," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1083–1092.

[5] H. Zha, J. Shen, K. Li, W. Greiff, M. T. Vanni, J. Han, and X. Yan, "Fts: Faceted taxonomy construction and search for scientific publications," 2018.

[6] K. Li, P. Zhang, H. Liu, H. Zha, and X. Yan, "Poqaa: Text mining and knowledge sharing for scientific publications," 2018.

[7] K. Li, W. Lu, S. Bhagat, L. V. Lakshmanan, and C. Yu, "On social event organization," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 1206–1215.

[8] K. Li, Y. He, and K. Ganjam, "Discovering enterprise concepts using spreadsheet tables," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1873–1882.

[9] K. Li, H. Zha, Y. Su, and X. Yan, "Concept mining via embedding," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 267–276.

[10] K. Li and et. al., "Unsupervised neural categorization for scientific publications," in *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 2018, pp. 37–45.

[11] K. Li, S. Li, S. Yavuz, H. Zha, Y. Su, and X. Yan, "Hiercon: Hierarchical organization of technical documents based on concepts," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 379–388.

[12] F. Tan, Y. Hu, C. Hu, K. Li, and K. Yen, "Tnt: Text normalization based pre-training of transformers for content moderation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 4735–4741.

[13] C. Yang, Y. Xiao, Y. Zhang, Y. Sun, and J. Han, "Heterogeneous network representation learning: A unified framework with survey and benchmark," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[14] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen *et al.*, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1235–1241, 2016.