A Secure Data Enclave and Analytics Platform for Social Scientists

Yadu N. Babuji, Kyle Chard, Aaron Gerow, Eamon Duede
Computation Institute
University of Chicago and Argonne National Laboratory
{yadunand,chard,gerow,eduede}@uchicago.edu

***Abstract*—Data-driven research is increasingly ubiquitous and data itself is a defining asset for researchers, particularly in the computational social sciences and humanities. Entire careers and research communities are built around valuable, proprietary or sensitive datasets. However, many existing computation resources fail to support secure and cost-effective storage of data while also enabling secure and flexible analysis of the data. To address these needs we present CLOUD KOTTA, a cloud-based architecture for the secure management and analysis of social science data. CLOUD KOTTA leverages reliable, secure, and scalable cloud resources to deliver capabilities to users, and removes the need for users to manage complicated infrastructure.CLOUD KOTTA implements automated, cost-aware models for efficiently provisioning tiered storage and automatically scaled compute resources.CLOUD KOTTA has been used in production for several months and currently manages approximately 10TB of data and has been used to process more than 5TB of data with over 75,000 CPU hours. It has been used for a broad variety of text analysis workflows, matrix factorization, and various machine learning algorithms, and more broadly, it supports fast, secure and cost-effective research.**

I. INTRODUCTION

Data is fast becoming a crucial, defining, asset for researchers. Entire fields, including those new to computational practices, are quickly embracing data-driven research. However, the increasing scale and complexity of analysis and the fact that datasets are often proprietary, or sensitive, creates unique new challenges. The centrality of data has inspired new processes that are designed for specific datasets, which typically results in tightly coupled environments that discourage reusability and agility. To support the needs of data-driven research, we developed CLOUD KOTTA[1], a unique cloud-based framework that enables the secure and cost-effective management and analysis of large, potentially sensitive datasets.

To address the growing reliance on data in research (particularly in the social sciences and humanities) scholars are increasingly replacing on-premise infrastructure with cloud-based solutions such as those offered by Amazon Web Services (AWS). This trend is not difficult to explain: cloud platforms provide high reliability, availability, and download performance without encumbering researchers with managing on-site infrastructure. The adoption of cloud-based services has also afforded new avenues for exploration. For example, when storage is co-located with elastic computing capacity with which data can be analyzed, aggregated, and integrated on-demand, researchers can take bigger risks and explore new analyses more flexibly. CLOUD KOTTA enables this kind of agility across fluid groups while also ensuring scalability, security, and data provenance.

Cloud-based infrastructure also has the advantage of helping centralize disparate teams. For example, given the sensitivity, value, and size of many datasets, it is often not feasible to replicate and download entire datasets for analysis on local computers or clusters. In many cases, circuitous approval procedures are necessary to gain access to data, and users must adhere to strict data-use agreements. Migrating data from the environment where it is hosted adds further complexity in this respect, and cloud-based strategies can exacerbate these challenges. So, when it comes to cloud-based infrastructure, it is important for researchers to develop a unified strategy. CLOUD KOTTA offers a strategy that is cost-effective, secure with respect to data policies, offers sustainable short- to long-term storage, and is scalable for demanding workloads.

CLOUD KOTTA is designed to address the requirements of two canonical use cases: managing community datasets securely and providing scalable compute resources. The first use case is motivated by a growing need for researchers to make valuable datasets available to certain research communities. This might be required by funding agencies or institutions, though, it is generally helpful to share data when establishing new research groups. The most important requirements for this use case are that CLOUD KOTTA be:

- Secure: Data should be stored securely and accessible only to authorized and authenticated users.
- Scalable: Data can be large, the storage system should scale to the data.
- Reliable: Data should be stored reliably, using backups in case of failure or corruption.
- Available: Data should be available to geographically distributed users.
- High performance: Large amounts of data should be moveable easily and quickly for analysis, download, and archival.
- Cost-effective: Costs associated with data storage should be minimized to encourage use.

The second use case is motivated by a large-scale movement towards data-intensive research. As data sizes grow and analyses become more computationally intensive, the

[1]Available at https://github.com/yadudoc/cloud_kotta.

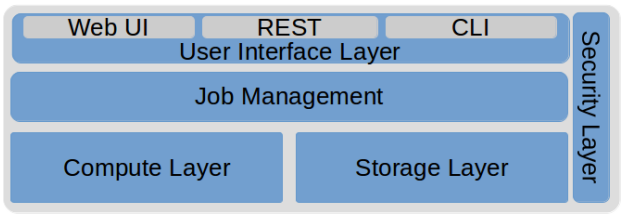

Fig. 1. Logical Architecture of CLOUD KOTTA

requirements often exceed the computational capabilities of individual researchers. As such, researchers need to be able to scale analyses from individual computers to distributed, parallel modes of computing. To address these priorities, CLOUD KOTTA should be:

- Secure: Authorizations should control what data can be analyzed and which analyses should be isolated.
- Scalable: Analyses should scale to data, exploit parallelism where possible, and leverage large scale computing infrastructure for efficient performance.
- Cost-effective: Costs should be comparable or lower than using local compute resources.
- Easy to use: Interfaces should make it simple to access the underlying infrastructure.

II. ARCHITECTURE & IMPLEMENTATION

The CLOUD KOTTA architecture is depicted in Fig. 1. The entire system is comprised of a web interface and REST API for accessibility; a set of event-based management and monitoring software to ensure reliable job execution; a compute layer that provides cost efficient compute resources; a fast and cost-effective storage layer; and an extensible and customizable security fabric that permeates all of the above components.

CLOUD KOTTA is designed to be deployed on *Amazon Web Services* (AWS), the ecosystem of its intended users. Where possible, CLOUD KOTTA leverages existing cloud services as they are scalable, reliable, secure, and cost-effective. The entire CLOUD KOTTA system is open source and can be deployed using a reproducible *CloudFormation* configuration which can be further customized to match target workloads.

A. User Interface

CLOUD KOTTA offers three interfaces: a web interface, a REST API, and a command line interface (CLI) accessible from the login node. This range of interfaces supports broad usage scenarios, enabling intuitive web access for web-based users and advanced programmatic and CLI support to facilitate customizable and automated invocation by technical users.

The supported interfaces support the same operations including, for example, browse datasets, upload new data, view and download results from previous analyses, submit and manage analyses. As with the entire CLOUD KOTTA architecture, the interfaces are secured, restricting access to authenticated, authorized users. The general architecture is centered around data stored in AWS Simple Storage Service (S3) buckets. Users can browse accessible data that they are permitted to access in S3, and they can also upload files to their own private S3 buckets. Once uploaded, files are available to be specified as inputs to submitted jobs. To support dynamic sharing scenarios, such as emailing colleagues the results of an analysis, CLOUD KOTTA provides support to construct short-term, anonymous URLs.

Submitting an analysis requires a description of the application (scripts, executables, etc), a list of inputs (S3, external URLs), a list of output files to be saved, and a maximum wall-time. In addition to supporting jobs with arbitrary executables, applications can be templated to create pipelines with simplified user interfaces. That is, other users wishing to reuse an analysis can simply complete a customized form with the required parameters. Users submit jobs via web forms in the Web interface, or by specifying the job as a JSON file for the CLI and REST interfaces.

B. Storage Layer

CLOUD KOTTA's storage layer uses a mix of AWS storage services that provide different guarantees regarding access time, durability, and availability with different cost models. The types of storage used by CLOUD KOTTA are:

- **Elastic Block Storage (EBS)**: a high performance block storage model that can be mounted as a file system on an EC2 instance.
- **S3 *standard***: a reliable object store that provides high performance access via HTTP(S).
- **S3 *infrequent access***: an object store with reduced storage cost at the expense of availability.
- **Glacier**: an archival storage model that provides high durability at a low price with high data retrieval times.

CLOUD KOTTA utilizes a caching model that is implemented using automated data life-cycle policies that manage data migration between storage tiers based on access patterns. Frequently accessed data resides on S3, as it is fast and highly available, whereas data that is accessed infrequently is moved to Glacier, as it provides durable, low cost storage at the expense of longer retrieval time. Fig. 2 illustrates the storage tiers used in CLOUD KOTTA's data model. Transferring data to lower tiers helps minimize the cost associated with providing high availability. The primary store for data in CLOUD KOTTA is S3. When data is analyzed, it can either be staged directly from S3 to ephemeral instance storage or EBS (which is subsequently mounted by an instance). Similarly, archived data stored in Glacier or S3 infrequent access buckets is staged to S3 when needed before being staged for analysis. Outputs are staged back to S3, guaranteeing durability.

CLOUD KOTTA's data model has important advantages over a static storage configuration. While EBS provides low-latency access, it is more than three times as expensive as S3-Standard and, in addition, must be mounted as a file system on a live machine to access data. By storing data in S3, a small overhead is incurred to stage data for compute, however, this latency is nominal in most cases and comprises a fraction of the time it takes to provision and execute a job. In addition to the cost

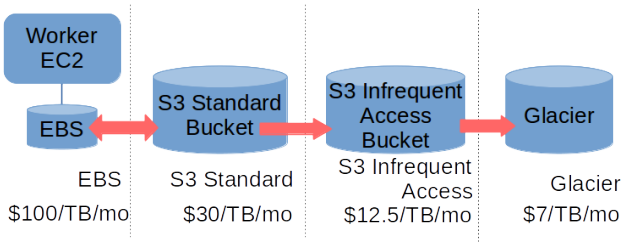

Fig. 2. Storage tiers in CLOUD KOTTA and the heuristics used to minimize storage costs. Some costs may differ across regions and configurations.

benefits, S3 provides support for rich access control features, as well as the ability to publish and access data directly via HTTP.

C. Compute Layer

The analyses for which CLOUD KOTTA is designed often comprise independent, long running, loosely coupled jobs. To support this class of workload, CLOUD KOTTA offers a scalable compute layer built upon elastic pools of Elastic Compute Cloud (EC2) instances.

EC2 is a virtualized computing environment in which users can lease virtual machines (VM) with varying computational resources. Instances are organized by region and *Availability Zone* (AZ). Regions represent different geographic locations whereas AZs are datacenters located within a specific region. Pricing models differ between AZs and, due to the independent failure models, users can achieve high reliability by distributing applications across AZs. EC2 instances are provisioned according to a market model in which users pay for the resources consumed.

CLOUD KOTTA can be configured to use two different EC2 market models. **On-demand** instances are offered at a fixed hourly price where instances live until they are terminated by a user. **Spot** instances are offered using a dynamic price model where users "bid" a maximum hourly price and instances are terminated by AWS if the market price exceeds the user's bid. Spot markets are typically a fraction of the on-demand price.

CLOUD KOTTA is designed to support two classes of workloads: short development jobs requiring quick responses, with minimal compute resources, and longer running production tasks that are computationally intensive, but more tolerant of delays. To meet the needs of these two asymetric workloads, CLOUD KOTTA offers two independent pools of compute resources. The development pool is comprised of on-demand instances, with at least one instance accessible at all times. To minimize the cost of executing computationally intensive, long running, yet delay tolerant production workloads, we utilize spot instances. CLOUD KOTTA relies on an automated bidding model to provision resources across AZs (to avoid price fluctuations in an AZ). Administrators can configure the bidding model to use static or policy-based bid prices (some fraction of the equivalent on-demand price, for example).

While using spot instances can significantly reduce costs, instance revocations are inevitable. To mitigate the problems associated with instance termination, CLOUD KOTTA manages queues that ensure jobs that do not complete are resubmitted to the queue and executed again on a new instance. By provisioning spot instances on-demand, CLOUD KOTTA can meet the demands of what are often sporadic and bursty workloads while also helping minimize costs [1]. CLOUD KOTTA currently uses a pre-defined EC2 instance type for each of it's queues. In future work, we will integrate CLOUD KOTTA with cost-aware provisioning [2], [1] and profiling [3] approaches to improve the selection of instance types based on cost and execution time.

D. Job Management

CLOUD KOTTA uses a job management layer to control the execution of arbitrary user analyses on the compute layer. User submitted tasks include the input files, execution scripts, and output files. Users must also choose if the task is a development or production job. When tasks are submitted, the description is stored in a database such that it can be accessed by the job management layer and the instances executing the analysis. To execute the task, the job management layer determines the user's access permissions, associates the user's role with the description, and places the job in the appropriate queue.

We leverage a queue model as it provides a reliable method for distributing jobs across a pool of EC2 instances. Worker nodes (EC2 instances) poll the queue for waiting tasks. If a task is available the worker moves it from a pending queue to the active queue. This active queue is used to manage execution and ensure that no tasks are lost. The worker retrieves the task description from the database and begins execution. In the case where spot instances are used, we must account for unreliability of the underling infrastructure. To do so, the job management layer includes a monitoring service that periodically checks instance health (e.g., for termination or other failures). If instances are terminated, the monitoring service will resubmit the task to the pending queue. Throughout execution, the worker node writes job status markers to the database. This information provides worker statistics (CPU, I/O and RAM utilization) and job progress, both of which are accessible to the user to monitor job execution. Upon completion, the worker will stage output data to S3, and update the database with the completion code of the task.

E. Security

The final layer of CLOUD KOTTA is the security fabric that permeates the system. CLOUD KOTTA uses Amazon's OAuth 2 model (*Login with Amazon*) for authentication. Users are able to login using their Amazon credentials and CLOUD KOTTA is therefore not responsible for managing user passwords. Before being granted access to the system, users must first be registered in CLOUD KOTTA's database and given an appropriate role. When users login using the OAuth 2 workflow, CLOUD KOTTA is given a short-term delegated access token. This token can be used to retrieve information about the user from Amazon as well as to use services as the user.

CLOUD KOTTA is built around a role-based access control model in which users are assigned roles, for example *kotta-public-only* and *kotta-read-WOS-private*, where *WOS* refers to the private Web of Science dataset. Policies associated with roles define permissions for specific resources (e.g., data access in S3). Given that all access is controlled by roles, worker nodes must assume a role before they can access restricted data. Other CLOUD KOTTA services are also given appropriate privileges by internal roles such as *web-server*, *task-executor*. These roles, unlike user roles, have access to the internal database, queues, notification systems and are capable of controlling scaling functionality. This role based access model limits validity of credentials to a small window limiting the risk of exposing valuable long-lived credentials. To adhere with the principle of least privilege, CLOUD KOTTA users are initially given no roles or privileges. They are incrementally granted permissions when required.

Data authorization and access control is implemented on S3 buckets. By default, access is not permitted unless it is explicitly granted via a policy. S3 buckets are associated with policies that prescribe permissions. Policies are then associated with roles that give permissions to users. The data stored on S3 buckets are server-side encrypted and accessible only from a Virtual Private Cloud (VPC) Endpoint. This guarantees that traffic between the S3 bucket and the compute instances remain private.Output data is also stored in S3. It is initially created as a private object only accessible to the creator. As CLOUD KOTTA is used by collaborating groups of users, it is important that data can be shared. To provide this capability, we use short-term signed URLs, like those used by other Cloud services (e.g., Google Drive and DropBox) that provide short term access to the holder of the URL.

CLOUD KOTTA implements a strong security model between instances and other services. The compute layer is hosted within a private subnet enclosed within a VPC. This ensures that compute instances are not directly accessible via the internet. Worker nodes are associated with a *task-executor* role that has few privileges (e.g., it cannot access any data) However, this is a trusted role that can be used to change to a user role for a short period of time. This is crucial as it enables a worker node, executing on behalf of a user, the ability to inherit the user's role and therefore access any data needed by the job. This approach ensures that even a running task is only able to access data for which the user is authorized to access. After staging data, the worker returns to the *task-executor* role to execute the job.

Finally, CLOUD KOTTA records every action performed by the system to ensure that data access and usage can be thoroughly audited. This information is recorded in a database such that administrators can export an audit log for any dataset, user, or service.

III. USAGE & APPLICATIONS

CLOUD KOTTA has been deployed and used over the past six months by a range of computational social scientists.

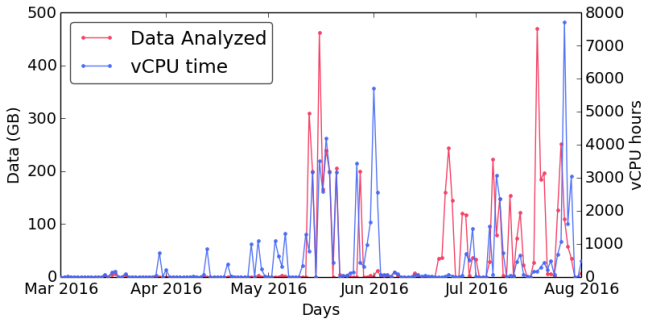

Fig. 3. Monthly usage of CLOUD KOTTA since deployment.

Active use cases for CLOUD KOTTA include text analysis, semantic word embedding, matrix factorization, optical character recognition, and social network analysis. Here we describe usage and illustrate some representative analyses.

A. Usage

CLOUD KOTTA has been used to develop, test and run a broad range of analytics on an array of datasets. CLOUD KOTTA currently manages datasets that collectively total nearly 10TB of data. It has been used for varied development cycles ranging from researchers running off-the-shelf tools on sample data, to teams of programmers developing and testing new methods on large, proprietary datasets. Throughout its development, CLOUD KOTTA was designed to speed up development and analysis cycles in a secure and flexible manner, while reducing costs induced by disparate and often idle compute resources. To-date, CLOUD KOTTA has been used to process over 5TB of data with over 75,330 CPU-hours. As our implementation of CLOUD KOTTA has solidified and become more robust, usage has grown (see Figure 3). The observed usage patterns affirm the choice of elastic cloud computing infrastructure as both data access and compute usage are particularly sporadic with peaks of over 7,000 compute hours in a single day and other days with none.

B. Applications

CLOUD KOTTA currently hosts a number of proprietary and sensitive datasets that have been used for a variety of workloads. CLOUD KOTTA has been primarily used to develop and run time-intensive text analyses, large scale matrix factorization, and optical character recognition (OCR). Other use cases have also been deployed with CLOUD KOTTA, but these three exemplify cases for which CLOUD KOTTA was designed: they are exploratory, large scale, and require private data that is shared among a strict set of users.

1) Text Analysis: One of the first use cases that was developed to run on CLOUD KOTTA is a tool-chain for text analysis. The analysis, designed to run on large collections of text, consists of four phases, each with separate inputs. The jobs consists of a series of pre- and post-processing scripts and wrapping natural language models. For these jobs, data is normalized and divided into logical bins, submitted to semantic analysis and post-processed to provide organized output.

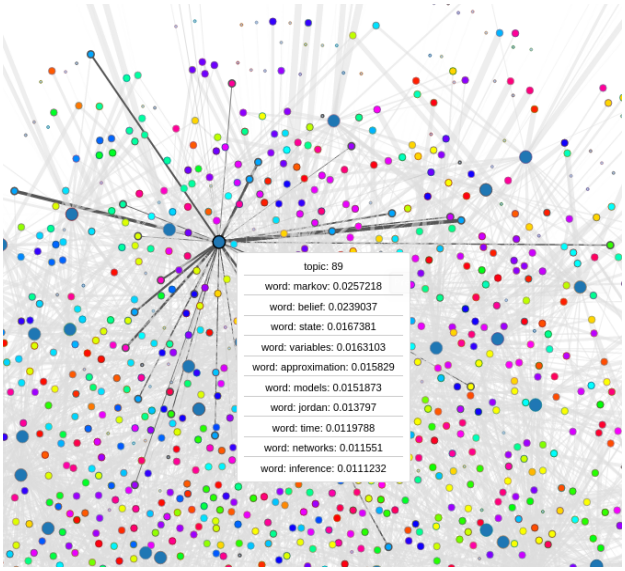

Fig. 4. Interactive analytics based on the analysis of researchers' publications. Shown is the author-to-topic network highlighting connections via topic #89, about Markov models.

The semantic model – the central analysis – includes doc2vec, word2vec [4] and various probabilistic topic models [5], [6]. These tend to be memory and compute intensive.

This workflow has unique challenges that can discourage exploration and slow development. It tends to involves free parameters at various phases, the scaling profile of many semantic models is unpredictable and the veracity of results is often measure qualitatively, making exploratory runs crucial. CLOUD KOTTA allowed our users to efficiently explore the parameter space to optimize a models' specification. One particular example is the Author-Topic (AT) Model [5], a probabilistic graph model that fits distributions of words, documents and authors to topics observed in texts. With CLOUD KOTTA, a multi-dimensional grid search was performed to assess the quality of various specifications attenuating the number of topics, and various fitting parameters that affect the interpretability of topics. Each run took approximately three days and 16GB of RAM. The final outputs of these runs were used to develop an interactive platform for researchers (Fig. 4) to generate explore commonalities and to propose future collaborators based on their existing work [7].

2) Matrix Factorization: Another use case where CLOUD KOTTA has been used was in developing a new method of multiple imputation (MI). When faced with lossy data, researchers often use MI to fill in missing values. Traditional MI establishes a missingness pattern on a given response which is then used in a regression with non-missing responses as parameters. The "multiple" aspect of MI is that after being imputed, new values can be used to increase the accuracy of imputing still-missing values. MI is cross-validated to assess stability and provide error-bounds. As a result, MI is computationally intensive and imposes strong statistical assumptions. A more fundamental weakness of parametric MI, however, is that it disregards latent structure in the response matrix. Low rank and low norm matrix factorization offer alternatives to parametric MI that can exploit patterns throughout the response structure [8].

In developing and testing low rank and low norm alternatives to MI, CLOUD KOTTA was used to run a range of tests simultaneously. These models were implemented in *Julia*, a relatively new scientific programming language. Because there is a stochastic component to low rank and low norm models, cross-validation on a held-out set of data was used to evaluate results over many random folds. Once the models were developed, CLOUD KOTTA was able to execute a large batch of validation jobs to provide pooled results. In these sets, a single run on a 2,500 by 122 matrix used 32 cores on a single instance and 100 GB of RAM for 10 hours. This made CLOUD KOTTA's ability to run multiple jobs in parallel a crucial feature over the course of development.

3) Optical Character Recognition: A third application for which CLOUD KOTTA has been used in production is OCR. OCR is the process of extracting text, figures, tables, and other features from rasterized images of documents. OCR is effectively a kind of object recognition that relies on trained models of character classification – models that are computationally intensive. CLOUD KOTTA was used to run OCR software on over 10 thousand grant proposals and scholarly texts. Processing these documents required 20 hours using 10, 32-core instances and 75GB of RAM. With CLOUD KOTTA , what would have taken over a month on personal hardware, was finished in a single day.

IV. RELATED WORK

Computational social science communities are investigating a broad range of approaches for hosting proprietary datasets and conducting scalable analytics. For example, researchers have used hybrid cloud models [9], extended common tools to analyze data at scale [10], and developed environments for securely analyzing data in controlled VMs [11]. CLOUD KOTTA is unique in its support for a wide range of data, a general architecture that accommodates many use cases, and by its automated, scalable storage and analytics environments.

Many scientific communities now have a broad range of data repositories available for storing and accessing different types of data (e.g., biomedicine [12], climate [13], and astronomy [14]). Systems are typically developed around a static data repository that requires significant administrative overhead to populate, curate and manage. Each has independent identity management sub-systems that have been developed to control access to data. But more importantly, most existing systems are static, isolated data environments, that provide minimal management capabilities separate from compute resources. Science gateways [15] aim to bridge this gap by abstracting the complexity of using large scale computing infrastructure. These systems typically provide access to shared datasets (e.g., in a repository) and resources through high level interfaces (workflows, portals, etc.). Examples of

commonly used gateways include CyberGIS [16] for geoscience and iPlant [17] for ecology. Most science gateways are built on more traditional High Performance Computing (HPC) infrastructure. However, recent work has focused on cloud-based solutions [18], [19]. CLOUD KOTTA acts as a fabric on which next generation data repositories and cloud-hosted gateways could be developed in a domain-agnostic setting.

CLOUD KOTTA can deploy customized, cloud-based clusters similar to a number of other systems. For example, CloudMan [20] and StarCluster [21] allow uses to deploy clusters for hosting and executing workflows. These systems, and others, are designed to aid the creation of clusters for semi-permanent usage. Other systems, such as Globus Galaxies [19] and Makeflow [22], enable on-demand and elastic cluster provisioning in response to workload. CLOUD KOTTA is unique, however, in its use of commodity AWS services and its broad focus on providing a framework for secure data storage and analysis.

V. SUMMARY

CLOUD KOTTA provides a secure and scalable data enclave and analytics environment for computational and data-drive social sciences. It addresses a gaping hole in the current infrastructure available to researchers, providing a model via which, even resource limited researchers can gain access to scalable data storage, elastic computing capacity, and cutting edge analysis algorithms without deploying and operating their own infrastructure. Moreover, it provides these capabilities while also optimizing performance and cost using automated data management and compute provisioning techniques.

In the six months since deployment CLOUD KOTTA has quickly grown to host a dozen private datasets (e.g., IEEE, Web of Science, and ACM) as well as several public datasets (e.g., US patents and Wikipedia). It has been used by dozens of researchers, students, and teachers to perform a wide variety of text analysis, machine learning, image recognition, and network analysis algorithms.

Our future work focuses on building an ecosystem around CLOUD KOTTA by developing a suite of data analytics frameworks from which users can more easily conduct analyses. We will continue to engage social scientists to add datasets to the system while looking to extend its capabilities to other disciplines. We are particularly interested in further developing algorithms for improving data lifecycle and compute management to better meet the needs of users with respect to performance, time, and cost.

ACKNOWLEDGMENTS

The authors thank Nandana Sengupta, Nathan Bartley, and Cha Chen for developing applications on CLOUD KOTTA . This research was supported by grants from the John Templeton Foundation to the Metaknowledge Research Network, IBM for Computational Creativity, and a gift from Facebook.

REFERENCES

[1] R. Chard, K. Chard, K. Bubendorfer, L. Lacinski, R. Madduri, and I. Foster, "Cost-aware cloud provisioning," in *Proceedings of the 11th International Conference on e-Science*, August 2015, pp. 136–144.

[2] R. Chard, K. Chard, K. Bubendorfer, L. Lacinski, R. Madduri, and I. Foster, "Cost-aware elastic cloud provisioning for scientific workloads," in *Proceedings of the 8th International Conference on Cloud Computing (CLOUD)*, June 2015, pp. 971–974.

[3] R. Chard, K. Chard, B. Ng, K. Bubendorfer, A. Rodriguez, R. Madduri, and I. Foster, "An automated tool profiling service for the cloud," in *Proceedings of the 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, May 2016, pp. 223–232.

[4] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[5] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth, "The author-topic model for authors and documents," in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*. AUAI Press, 2004, pp. 487–494.

[6] J. Zhang, A. Gerow, J. Altosaar, J. Evans, and R. J. So, "Fast, flexible models for discovering topic correlation across weakly-related collections," in *Proceedings of Empirical Methods in Natural Language Processing*, 2015.

[7] A. Gerow, B. Lou, E. Duede, and J. Evans, "Proposing ties in a dense hypergraph of academics," in *Social Informatics*. Springer, 2015, pp. 209–226.

[8] M. Udell, C. Horn, R. Zadeh, and S. Boyd, "Generalized low rank models," *arXiv preprint arXiv:1410.0342*, 2014.

[9] S. Abramson, W. Horka, and L. Wisniewski, "A hybrid cloud architecture for a social science research computing data center," in *Proceedings of the 34th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, June 2014, pp. 45–50.

[10] M. A. Saleem, B. Varghese, and A. Barker, "Bigexcel: A web-based framework for exploring big data in social sciences," in *Proceedings of the IEEE International Conference on Big Data (Big Data)*, Oct 2014, pp. 84–91.

[11] J. Zeng, G. Ruan, A. Crowell, A. Prakash, and B. Plale, "Cloud computing data capsules for non-consumptiveuse of texts," in *Proceedings of the 5th ACM Workshop on Scientific Cloud Computing (ScienceCloud)*, 2014, pp. 9–16.

[12] M. Mailman, M. Feolo, Y. Jin, M. Kimura, K. Tryka, R. Bagoutdinov, L. Hao, A. Kiang, J. Paschall, L. Phan, N. Popova, S. Pretel, L. Ziyabari, M. Lee, Y. Shao, Z. Wang, K. Sirotkin, M. Ward, M. Kholodov, K. Zbicz, J. Beck, M. Kimelman, S. Shevelev, D. Preuss, E. Yaschenko, A. Graeff, J. Ostell, and S. Sherry, "The NCBI dbGaP database of genotypes and phenotypes." *Nature Genetics*, vol. 39, no. 10, pp. 1181–1186, 2007.

[13] "National Climatic Data Center (NCDC)," http://www.ncdc.noaa.gov/, web site. Accessed: May, 2016.

[14] "SIMBAD Astronomical Database," http://simbad.u-strasbg.fr/simbad/, web site. Accessed: May, 2016.

[15] N. Wilkins-Diehr, "Special issue: Science gatewayscommon community interfaces to grid resources," *Concurrency and Computation: Practice and Experience*, vol. 19, no. 6, pp. 743–749, 2007.

[16] Y. Liu, A. Padmanabhan, and S. Wang, "CyberGIS gateway for enabling data-rich geospatial research and education," in *Proceedings of the IEEE International Conference on Cluster Computing (CLUSTER)*, Sept 2013, pp. 1–3.

[17] D. Stanzione, "The iPlant collaborative: Cyberinfrastructure to feed the world," *Computer*, vol. 44, no. 11, pp. 44–52, Nov 2011.

[18] W. Wu, H. Zhang, Z. Li, and Y. Mao, "Creating a cloud-based life science gateway," in *Proceedings of the 7th IEEE International Conference on e-Science*, Dec 2011, pp. 55–61.

[19] R. Madduri, K. Chard, R. Chard, L. Lacinski, A. Rodriguez, D. Sulakhe, D. Kelly, U. Dave, and I. Foster, "The Globus Galaxies platform: delivering science gateways as a service," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 16, pp. 4344–4360, 2015.

[20] E. Afgan, D. Baker, N. Coraor, H. Goto, I. M. Paul, K. D. Makova, A. Nekrutenko, and J. Taylor, "Harnessing cloud computing with galaxy cloud," *Nature Biotechnology*, vol. 29, pp. 972–974, 2011.

[21] "StarCluster," http://star.mit.edu/cluster/, web site. Accessed: May, 2016.

[22] M. Albrecht, P. Donnelly, P. Bui, and D. Thain, "Makeflow: A portable abstraction for data intensive computing on clusters, clouds, and grids," in *Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies*. ACM, 2012, pp. 1:1–1:13.