Runway: machine learning model experiment management tool

Jason Tsay, Todd Mummert, Norman Bobroff, Alan Braz, Peter Westerink, Martin Hirzel
IBM Research, Yorktown Heights, NY, USA
{jason.tsay,braz.alan}@ibm.com,{mummert,bobroff,peterw,hirzel}@us.ibm.com

ABSTRACT

Runway is a cloud-native tool for managing machine learning experiments and their associated models. The iterative nature of developing models results in a large number of experiments and models that are often managed in an ad hoc manner. Runway is a workflow and framework independent tool that centrally manages and maintains metadata and links to artifacts needed to reproduce models and experiments. Runway provides a web dashboard with multiple levels of visualizations to evaluate performance and enable side-by-side comparisons of models and experiments.

1 INTRODUCTION

Machine Learning (ML) models are increasingly at the core of applications and systems. The process around developing these models is highly iterative and experiment-driven [4]. The often non-linear and non-deterministic nature of implementing ML models [7] results in a large number of diverse models. Through interviews we find that data scientists tend to manage models using ad hoc methods such as notebooks, spreadsheets, file system folders, or PowerPoint slides. However, these ad hoc methods record the models themselves but not the higher-level *experiment*. For example, a data scientist developing a natural language classifier may wish to compare models from a support vector machine experiment to ones from neural networks. At best, extra effort must be spent to manage experiments and their models and at worst, effort is wasted on what one interviewee called “dead-end trials.” Given the increasing complexity and required computational time for ML models, reducing effort on experiments may greatly improve the efficiency of data scientists’ workflows. At the same time, data scientists also tend to work idiosyncratically with pipelines and workflows unique to the task at hand. This variety means that a “one size fits all” approach is insufficient.

To address these challenges, we introduce Runway, a prototype ML model experiment management tool with the following design goals: (1) multiple levels of model management, (2) workflow and framework independence, (3) visual tools for model evaluation and comparison, and (4) cloud-native architecture that allows for easy integration with existing platforms. Runway is currently internally available to data scientists at IBM. Runway’s design is also informed by a series of interviews with 27 data scientists at IBM from a wide variety of domains and by iterative agile development with sponsor users.

2 RELATED WORK

We position our work in a burgeoning field of engineering that assists in developing ML models and applications. Kim et al. [4] find through interviews that data scientists fulfill multiple important engineering roles towards connecting software systems to “real-world” data. They find something that we confirm in our own interviews: the sheer variety of titles, backgrounds, domains, and tasks for data scientists. An important commonality however between data scientists is familiarity with experiment-driven work or, as one interviewee put it, “I am used to designing experiments.” Patel et al. [8] find through interviews and studies with data scientists that the highly iterative and exploratory nature of developing ML models is a primary challenge. In particular, multiple aspects of the seemingly linear workflow interconnect and ML developers would waste time on dead-end experiments. They also find that for many tasks, evaluating performance is often more difficult than simply evaluating metrics.

Closely related to our work and model management tools are tools that support general ML engineering such as Gestalt [6] or TFX [1] and in particular model management tools such as ModelDB by Vartak et al. [9], ModelHub by Miao et al. [5], and MLModelScope[1]. Model management tools are concerned with indexing and tracking large numbers of ML models for future sharing, querying, and analysis. Such tools support data scientists in sensemaking and identifying insights for their models. Runway builds on model management tools by supporting multi-level management such as experiments and including visual evaluation tools.

3 IMPLEMENTATION

3.1 Architecture

Figure 1 shows the high-level architecture of Runway, which consists of three key components: (1) a REST API backend which is the core of the architecture, (2) a Software Development Kit (SDK) that allows data scientists to instrument their own Python 3 scripts, and (3) a web-based dashboard interface. Runway is also designed to be cloud-native and integrates easily with other services such as cloud object storage[2] and the IBM Deep Learning (DL) Service [2].

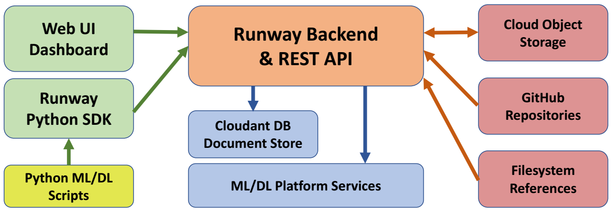


Figure 1: Runway high-level architecture.

Runway stores and organizes metadata about ML models in a hierarchy of Projects, Experiments, and Runs. *Projects* represent

SYSML’18, February 2018, Stanford, CA, USA
2018.

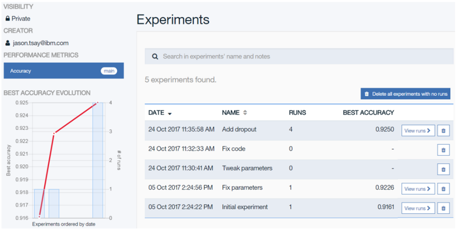
[1] http://mlmodelscope.org
[2] https://console.bluemix.net/catalog/services/object-storage

the task at hand such as image classification. They are made up of multiple *Experiments* which are approaches or ideas data scientists explore for the task, such as algorithms or network topologies. Each Experiment is made up of *Runs* which are specific managed models with their own set of parameters and metrics, and include all the *Artifacts* such as training data, code, and log files that are required to reproduce the model and result. Interviewees often considered *Experiments* as a particular code commit and each *Run* as a particular configuration and result from running this code. The API also manages *Credentials* to authenticate to external services which host the artifacts, including cloud storage, model training environments, and code hosting platforms such as GitHub. Each "layer" of the hierarchy has its own set of visualizations that assist in aggregating and making sense of the lower layer.

We found from interviews that most data scientists prefer using Python for developing ML models. However, due to a wide variety of frameworks and libraries and unique workflows, we decided that a workflow and framework independent approach was necessary. We provide a Python SDK and ask data scientists to instrument their existing Python scripts using this library. The SDK provides wrappers for most API calls and convenience functions such as uploading single files, folders, or archives as artifacts for a particular model. Instrumenting is straightforward, a data scientist copies boilerplate code into their existing script and adds specific modifications such as metrics and artifacts to track.

3.2 Model Evaluation Visualizations

The web-based dashboard is the main visual user interface for Runway. On top of managing models and their metadata, this dashboard provides visualizations to assist in understanding and evaluating parts of the ML model development process. Runway provides visualizations that summarize performance at multiple levels such as a Project-level line and bar chart that displays performance metrics for each Experiment in the Project and the number of Runs per Experiment in Figure 2. Visually aggregating multiple levels helps to make sense of the entire ML task [6]. This chart allows data scientists to track performance trends across experiments. The dashboard also provides visualizations to better understand relationships between hyperparameters and performance metrics through a scatter plot on the Experiment level as seen in Figure 3.



VISIBILITY
Private
CREATOR
jason.tsay@ibm.com
PERFORMANCE METRICS
Accuracy main

Experiments

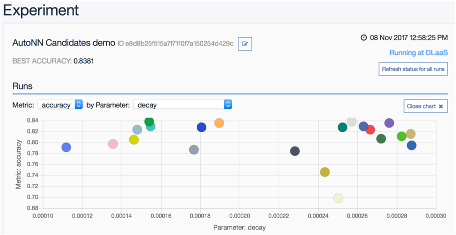
5 experiments found.

DATE	NAME	RUNS	BEST ACCURACY
24 Oct 2017 11:35:58 AM	Add dropout	4	0.9250
24 Oct 2017 11:32:33 AM	Fix code	0	-
24 Oct 2017 11:30:41 AM	Tweak parameters	0	-
05 Oct 2017 2:24:56 PM	Fix parameters	1	0.9226
05 Oct 2017 2:24:22 PM	Initial experiment	1	0.9161

Figure 2: Project-level metrics line and bar chart for all Experiments.

3.3 Model Comparisons

Another primary feature for evaluating models that the dashboard provides is the comparison of models. Once a data scientist selects two Runs for comparison, their metadata, performance metrics, hyperparameters, and artifacts are all displayed side-by-side. The comparison also provides features towards better understanding differences between models. For example, a difference in performance may be drilled down to a difference in how the model was configured. The dashboard assists in this process by Git-style highlighting of differences in text files such as scripts and configuration files and side-by-side display of visualizations such as learning curves. Both are visible in Figure 4.



Experiment

AutoNN Candidates demo ID e8d8b25f515a7f7110f7a150254d429c

BEST ACCURACY: 0.8381

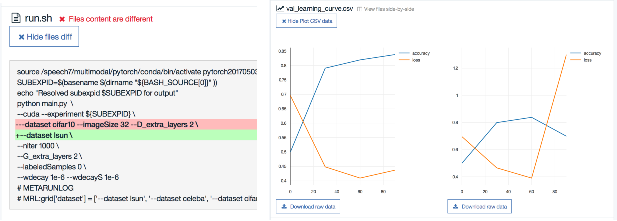
08 Nov 2017 12:58:25 PM

Running at DLaaS

Runs

Metric: accuracy by Parameter: decay

Figure 3: Experiment-level metric vs. hyperparameter scatter plot for all Runs.



run.sh ✖ Files content are different

```
source /speech7/multimodal/pytorch/conda/bin/activate pytorch2017050
SUBEXPID=$(basename $(dirname "${BASH_SOURCE[0]}" ))
echo "Resolved subexpid $SUBEXPID for output"
python main.py  \
--cuda --experiment ${SUBEXPID} \
---dataset cifar10 --imageSize 32 --D_extra_layers 2 \
+--dataset lsun \
--niter 1000 \
--G_extra_layers 2 \
--labeledSamples 0 \
--wdecay 1e-6 --wdecayS 1e-6
# METARUNLOG
# MRL:grid['dataset'] = ['--dataset lsun', '--dataset celeba', '--dataset cifa
```

val_learning_curve.csv

(a) Configuration files

(b) Performance

Figure 4: Comparing two experiment runs.

3.4 Preliminary Usage

Though still early, we have found anecdotally that Runway is especially useful for managing large numbers of automatically-generated models and identifying trends within these models. The lightweight and cloud-native nature of Runway also allows it to easily work with other ML and AI tools within IBM. For example, we have successfully used Runway in a full ML toolchain that also uses the IBM Deep Learning Service [2] and an automated parameter search service [3].

4 CONCLUSIONS

This paper describes our prototype ML model experiment management tool Runway. This prototype is currently available for data scientists within IBM with user studies ongoing. In the future, we expect that improving how data scientists perform and manage ML experiments will also improve related aspects of the iterative model development process. For example, our tool allows for uploading and archiving all artifacts related to training a particular model. Not only does this enable reproducibility and provenance for a particular experiment, the centralized location also allows for easy sharing of experiments with collaborators.

REFERENCES

[1] Denis Baylor, Eric Breck, Heng-Tze Cheng, Noah Fiedel, Chuan Yu Foo, Zakaria Haque, Salem Haykal, Mustafa Ispir, Vihan Jain, Levent Koc, Chiu Yuen Koo, Lukasz Lew, Clemens Mewald, Akshay Naresh Modi, Neoklis Polyzotis, Sukriti Ramesh, Sudip Roy, Steven Euijong Whang, Martin Wicke, Jarek Wilkiewicz, Xin Zhang, and Martin Zinkevich. 2017. TFX: A TensorFlow-Based Production-Scale Machine Learning Platform. In *Conference on Knowledge Discovery and Data Mining (KDD)*. 1387–1395. https://doi.org/10.1145/3097983.3098021

[2] B. Bhattacharjee, S. Boag, C. Doshi, P. Dube, B. Herta, V. Ishakian, K. R. Jayaram, R. Khalaf, A. Krishna, Y. B. Li, V. Muthusamy, R. Puri, Y. Ren, F. Rosenberg, S. R. Seelam, Y. Wang, J. M. Zhang, and L. Zhang. 2017. IBM Deep Learning Service. *IBM Journal of Research and Development* 61, 4 (July 2017), 10:1–10:11. https://doi.org/10.1147/JRD.2017.2716578

[3] Gonzalo I. Diaz, Achille Fokoue-Nkoutche, Giacomo Nannicini, and Horst Samulowitz. 2017. An effective algorithm for hyperparameter optimization of neural networks. *IBM Journal of Research and Development* 61 (July 2017), 9:1–9:11. Issue 4. https://doi.org/10.1147/JRD.2017.2709578

[4] Miryung Kim, Thomas Zimmermann, Robert DeLine, and Andrew Begel. 2016. The Emerging Role of Data Scientists on Software Development Teams. In *International Conference on Software Engineering (ICSE)*. 96–107. http://doi.acm.org/10.1145/2884781.2884783

[5] Hui Miao, Ang Li, Larry S. Davis, and Amol Deshpande. 2016. ModelHub: Towards Unified Data and Lifecycle Management for Deep Learning. *CoRR* abs/1611.06224 (2016). https://arxiv.org/abs/1611.06224

[6] Kayur Patel, Naomi Bancroft, Steven M. Drucker, James Fogarty, Andrew J. Ko, and James Landay. 2010. Gestalt: Integrated Support for Implementation and Analysis in Machine Learning. In *Symposium on User Interface Software and Technology (UIST)*. 37–46. http://doi.acm.org/10.1145/1866029.1866038

[7] Kayur Patel, James Fogarty, James A. Landay, and Beverly Harrison. 2008. Examining Difficulties Software Developers Encounter in the Adoption of Statistical Machine Learning. In *Conference on Artificial Intelligence (AAAI)*. 1563–1566. https://aaai.org/Papers/AAAI/2008/AAAI08-263.pdf

[8] Kayur Patel, James Fogarty, James A. Landay, and Beverly Harrison. 2008. Investigating Statistical Machine Learning As a Tool for Software Development. In *Conference on Human Factors in Computing Systems (CHI)*. 667–676. http://doi.acm.org/10.1145/1357054.1357160

[9] Manasi Vartak, Harihar Subramanyam, Wei-En Lee, Srinidhi Viswanathan, Saadiyah Husnoo, Samuel Madden, and Matei Zaharia. 2016. ModelDB: A System for Machine Learning Model Management. In *Workshop on Human-In-the-Loop Data Analytics (HILDA)*. 14:1–14:3. http://doi.acm.org/10.1145/2939502.2939516