Flare is a [Leverages](#) [Achieves](#) [Deploying cutting edge machine learning (ML) models at scale requires ingesting data from various sources. Today, this data preprocessing is typically implemented using separate tools like [Pandas](#) or [Spark](#) that then feed into [PyTorch](#) or [TensorFlow](#). While these ML systems go to great lengths to optimize the performance of their ML kernels, large amounts of performance are lost when data needs to move from one framework to another.

### Motivation
- Deploying cutting edge machine learning (ML) models at scale requires ingesting data from various sources.
- Today, this data preprocessing is typically implemented using separate tools like Pandas or Spark that then feed into PyTorch or TensorFlow.
- While these ML systems go to great lengths to optimize the performance of their ML kernels, large amounts of performance are lost when data needs to move from one framework to another.

### Flare
- Flare is a query accelerator built for Apache Spark
- Achieves order of magnitude speedups on DataFrame and SQL workloads
- Flare compiles optimized query plans generated by Catalyst (in-built query optimizer for Apache Spark) to native code
- Leverages Lightweight Modular Staging (LMS) to generate native code

### Lantern
- Lantern is a highly expressive machine learning framework written in Scala
- Attains the performance of "define-then-run" machine learning frameworks like Tensorflow while preserving expressiveness of "define-by-run" frameworks such as PyTorch
- Backpropagation is implemented using functions with callbacks; where forward pass is executed as a sequence of function calls and the backward pass with the corresponding function returns (uses CPS in particular)
- Leverages Lightweight Modular Staging (LMS) to generate low-level (C/CUDA) code

### Experiments
- Prior experiments have shown promising results on using Flare in conjunction with Tensorflow
- For a simple case where Tensorflow classifier is used as a Spark UDF, Flare produced over 1,000,000x speed up for some cases

---

**Figure 1:** Flare system overview

**Figure 2:** Performance comparison of Postgres, HyPer, Spark SQL, Flare in TPC-H (SF10) (from: Essertel et al OSDI'18)

**Figure 3:** Running time of SqueezeNet and ResNet50 for different frameworks (from: Fei et al ICFP '19)

**Figure 4:** Flare + Lantern system overview

**Figure 5:** Running time (ms) of a query containing a TF classifier as an UDF (from: Essertel et al OSDI '18)

**Figure 6:** Total Running time (ms) for a basic regression task (Fuel efficiency prediction) taken from TF documentation