Performance Prediction for Data Transfers in LCLS Workflow

Mengtian Jin[*]
Stanford University
Stanford, CA
mtjin@stanford.edu

Youkow Homma
Stanford University
Stanford, CA
yhomma@stanford.edu

Alex Sim
Lawrence Berkeley Nat'l Laboratory
Berkeley, CA
asim@lbl.gov

Wilko Kroeger
SLAC National Accelerator Laboratory
Stanford, CA
wilko@slac.stanford.edu

Kesheng Wu
Lawrence Berkeley Nat'l Laboratory
Berkeley, CA
kwu@lbl.gov

ABSTRACT

In this work, we study the use of decision tree-based models to predict the transfer rates in different parts of the data pipeline that sends experiment data from Linac Coherent Light Source (LCLS) at SLAC National Accelerator Laboratory (SLAC) to National Energy Research Scientific Computing Center (NERSC). The system monitoring the data pipeline collects a number of characteristics such as the file size, source file system, start time and so on, all of which are known at the start of the file transfer. However, these static variables do not capture the dynamic information such as current state of the networking system. In this work, we explore a number of different ways to capture the state of the network and other dynamic information. We find that in addition to using static features, using these dynamic features can improve the transfer performance predictions by up to 10-15%. We additionally study a couple of different well-known decision-tree based models and find that Gradient-Tree Boosting algorithm performs better overall.

CCS CONCEPTS

• **Networks** → **Network performance modeling**; **Network performance analysis**; • **Computing methodologies** → **Neural networks**; **Dimensionality reduction and manifold learning**; *Cluster analysis*; *Feature selection*; • **Applied computing** → *Physics*.

KEYWORDS

Network performance; Performance prediction; LCLS

ACM Reference Format:
Mengtian Jin, Youkow Homma, Alex Sim, Wilko Kroeger, and Kesheng Wu. 2019. Performance Prediction for Data Transfers in LCLS Workflow. In *Systems and Network Telemetry and Analytics (SNTA '19), June 25, 2019, Phoenix, AZ, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3322798.3329254

[*]The first two authors contributed equally to the paper.

ACM acknowledges that this contribution was authored or co-authored by an employee, contractor, or affiliate of the United States government. As such, the United States government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for government purposes only.
SNTA '19, June 25, 2019, Phoenix, AZ, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6761-5/19/06…$15.00
https://doi.org/10.1145/3322798.3329254

1 INTRODUCTION

Data collected from large scientific facilities nowadays are the major support to many scientific discoveries. The network that distributes the huge amount of data to scientists around world therefore plays a key role in this process. Being able to predict the performance of a large data pipeline could be critical to the resource planning and dynamic scheduling of the data transfers. In this work, we plan to predict the performance of a couple of stages of the data pipeline for moving a large amount of scientific data from Stanford to Berkeley (LCLS workflow). The prediction model proposed in this paper not only works for LCLS workflow, but also can be generalized and applied to other workflows.

Linac Coherent Light Source (LCLS) at SLAC National Accelerator Laboratory is a laser used to image molecules, which supports the fundamental studies of Chemistry, Biology, Physics and Technology. There are seven instruments with different detectors and X-ray beam characteristics that allow diverse types of experiments [4]. LCLS produces terebytes of data for each experiment, and the experiment data is then sent from SLAC to supercomputers at National Energy Research Scientific Computing Center (NERSC) to process the massive data quantities. The current data pipeline involves two stages, Fast Feedback (FFB) transfer and Analysis (ANA) transfer. FFB provides fast, low latency access to the collected data. It is only used by the active experiments. ANA is large in size (4PB), shared between all experiments and holds the experimental data for many months. The data flow in shown in Figure 1.

When data is generated from an experiment, it is written to files on DSS nodes via the data acquisition (DAQ). Multiple Data Storage Subnet (DSS) nodes are used in parallel for each data collection (a run) and files from the same node constitute a stream. Typically 5-6 streams are run in parallel. A data mover then transfers files to FFB storage from the DSS nodes. The data is then transferred from FFB to ANA. The typical FFB transfer rate is limited by the rate the file is written to a DSS node, usually around 100-200MB/s. The typical limit for the ANA transfer rate in this process is about 350-400MB/s due to the speed of checksum calculation. Somewhat frequently, the DSS to FFB transfer can also reach this checksum limit when the transfer is late and the file has already been written to the DSS node by the DAQ.

In this work, our aim is to use historical data from the LCLS workflow to build a model that can predict transfer rates of future transfers from DSS nodes to FFB and FFB to ANA. In particular, we conduct our analysis and model-building on a set of 258,765

Instrument	Count
cxi	28900
xpp	23110
mec	22509
xcs	16307
sxr	12354
mfx	9713
amo	8341

Table 1: Number of Records Collected by Each Instrument

transfers from May 2017 to January 2018 where 131,274 transfers take place from DSS to FFB and the remaining 127,491 are from FFB to ANA.

Yang et al. [4] worked on the same set of monitoring data as this work. Their work primarily uses the static features, and predictions were demonstrated to be accurate enough for detecting extremely slow file transfers, where the actual observed performance is significantly worse then predicted *normal* performance.

In this work, we seek to improve the prediction accuracy by including dynamic features. The bulk of this work is on extracting these dynamic features. Additionally, we will also be examining a number of decision-tree based techniques that are known to be able to make accurate predictions. By extracting dynamic features and using more accurate prediction techniques, we are able to make much more accurate predictions for all file transfers instead of just the expected transfer speed good enough for detecting extremely slow transfers.

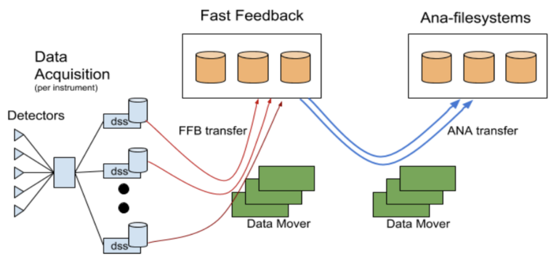

Figure 1: LCLS Data flow: Red line is DSS -> FFB transfer, Blue line is FFB -> ANA transfer

2 UNDERSTANDING THE RAW DATA

LCLS has seven instrumental stations with different types of detectors. The stations are distributed in two buildings known as the Near Experiment Hall (NEH) and the Far Experiment Hall (FEH), see Table 2. Due to this physical separation, these instruments are also attached to different local file systems. Table 1 has the number of files produced by each of the seven instruments used in the data collection. From this table we see that cxi, xpp and mec are used much more than the others. The overview of statistics on transfer rate and file size in FFB and ANA transfers are shown in Figure 2.

Next, we describe the file generation and transport process to give more information about what features are important to predicting the file transfer performance.

STATS	size (gigabyte)	transfer rate(MB/s)
median	4.0	47.9
mean	13.8	81.2
std	22.1	91.0
min	0	0
max	1304.14	498.2

(a)

STATS	size (gigabyte)	transfer time (MB/s)
median	4.0	342.7
mean	13.29	301.8
std	21.49	109.7
min	0	0
max	1304.14	522.2

(b)

Figure 2: Basic Statistics Summary of Transfer Rate and File Size on FFB and ANA. (a) DSS to FFB (b) FFB to ANA: 12 records which have file sizes over 1 TB due to configuration errors, and 76 transfers with either a file size or transfer rate of zero are removed from the analysis.

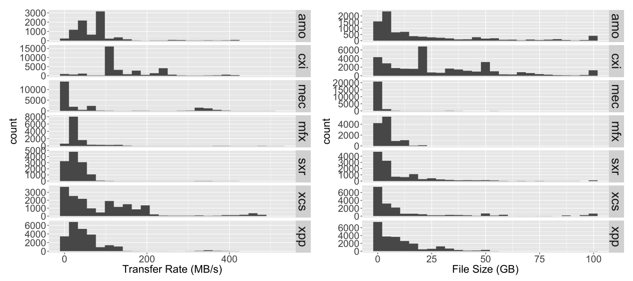

Figure 3: Left: Histogram of transfer rate (to FFB) by instrument. We see that on the whole, 'cxi' and 'amo' tend to have faster transfer rates while 'mec' has the slowest transfer rate. Right: Histogram of file size by instrument.

File Size. Based on past experience, we know that the performance of a file transfer is heavily dependent on the file size. The first feature we plan to explore is the file size. Figure 3 shows the distribution of the file sizes and transfer rates. We see that different instruments have clearly distinctive distributions.

Nodes and Streams. As data is generated from an experiment, it is written in parallel streams with each stream assigned to one DSS node. This data is then simultaneously written to the FFB by the data mover/host. Once a stream reaches a cap of 100 GB, the file is closed and a new stream is started. We call a set of data transfers a chunk. The structure of chunks and streams are embedded in the file name and have to be extracted. Such information allows us to better understand the data transport process and make more accurate predictions.

There are typically between 4-5 streams with a range from 1 to 6 streams. In Figure 4, we show the distribution of the median transfer rate of the streams in an experiment batch for the instruments in the

Instrument	FFB File System	DSS ⟶ FFB Host	FFB ⟶ ANA Host
amo, sxr, xpp	ffb11	psana{102,103}	psana{102,103}, psexport{01,02,05,06,07,08}
cxi, mec, mfx, xcs	ffb21	psana{201,202,203}	psana{201,203}, psexport{01,02,05,06,07,08}

Table 2: Summary of LCLS detectors: instruments in the top row are located in Near Experiment Hall (NEH) and those in the bottom row are located in the Far Experiment Hall (FEH).

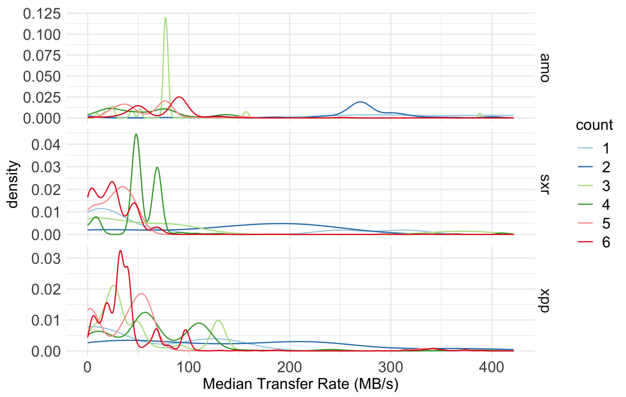

Figure 4: Probability densities for the median transfer rate of the streams in each experiment batch for NEH instruments.

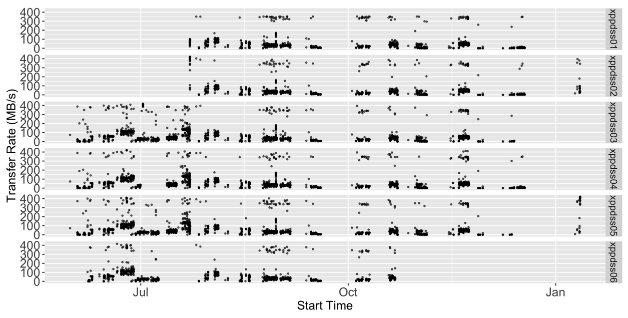

Figure 5: Scatter plot of start time vs transfer rate split by node for experiments on 'xpp'.

NEH. As we expected within a certain capacity, the more streams an experiment has, the slower the transfer rate tends to be. On top of looking at the number of streams, we can also look at how these streams are distributed to different nodes. In Figure 5, we plot the transfer rate over time for all nodes of instrument 'xpp,' and we can see that different nodes are dormant and active during different parts of the year.

Delayed Transfers. Different streams of a chunk are transferred at the same time, but there are instances where a stream's start time is significantly later than the start time of the first stream in the same chunk. One example of this is in Table 3 from chunk 00 of experiment 'e991-r0002' on instrument 'mfx'. We see that the first two transfers are almost 3 hours before the later two streams. The difference in target hosts suggests that there may have been a problem with psana203 during this time. A slightly more nuanced

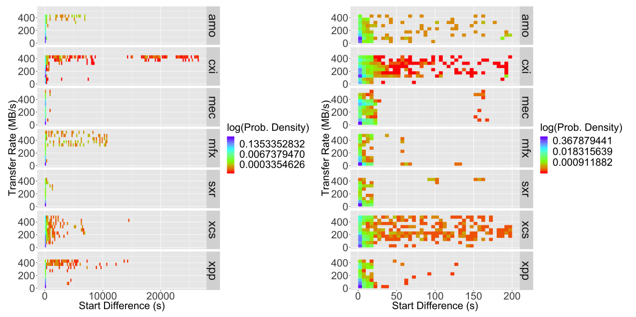

Figure 6: Impact of the start time difference on transfer rate. Left: Log density plot of start time difference vs transfer rate for all files. Right: Log density plot of start time difference vs transfer rate with time difference less than 200 seconds.

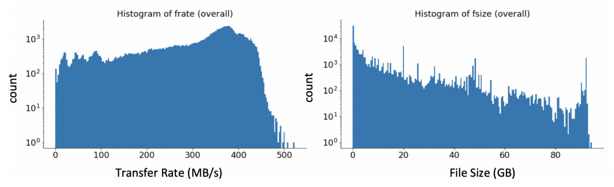

Figure 7: Distribution of Transfer Rate and File Size in ANA Transfer: Mean and median of transfer rate are 300.5MB/s and 341.7MB/s respectively. Mean and median of file size are 1.33GB and 3.95GB respectively. The count is in logarithm scale.

example is in Table 4. We see that the first two streams start at the same time, then there is about a 2 minute gap in subsequent streams. This leads to a slight bump in transfer rate. Similar to above, we see that the target host is different between the first two streams and the latter three.

From Figure 6, we clearly see the impact of the start time difference, especially when the start time difference is large. On the other hand, if the difference is on the order of a few minutes, then the pattern is less clear.

Transfers to ANA. ANA transfers share some similar patterns as shown above. However FFB is only used by the active experiments, whereas ANA holds experimental data for several months, so we also expect some different behaviors in ANA transfer. The overall distribution of file transfer rate and file size in ANA process is shown Figure 7.

The transfer also behaves differently depending on which instrument was used to collect the data as we can see from the distribution plots of transfer rate on each instrument in Figure 8.

Start Time	Stop Time	File Size (GB)	Transfer Rate (MB/s)	Target Host	Node
2017-06-21 16:02:02	2017-06-21 16:02:26	0.3168954	13.73635	psana201	mfxdss02
2017-06-21 16:02:02	2017-06-21 16:02:26	0.3168954	13.73635	psana201	mfxdss01
2017-06-21 18:48:26	2017-06-21 18:48:28	0.3168954	399.08803	psana203	mfxdss05
2017-06-21 18:48:26	2017-06-21 18:48:28	0.3168954	399.08803	psana203	mfxdss06

Table 3: Two streams from the same chunk are transferred more than two hours later than other streams, which seems to cause significant difference in transfer rates.

Start Time	Stop Time	File Size (GB)	Transfer Rate (MB/s)	Target Host	Node
2017-09-22 19:48:26	2017-09-22 19:56:57	98.79810	193.4623	psana201	xcsdss03
2017-09-22 19:48:26	2017-09-22 19:56:57	98.80620	193.7768	psana201	xcsdss02
2017-09-22 19:50:42	2017-09-22 19:56:58	98.80619	263.1926	psana202	xcsdss05
2017-09-22 19:53:31	2017-09-22 19:59:45	98.70099	264.1363	psana202	xcsdss06
2017-09-22 19:55:07	2017-09-22 20:01:01	98.81428	279.6552	psana202	xcsdss04

Table 4: The last stream of this chunk has a minor delay of about two minutes, which seems to have a smaller impact on transfer rate.

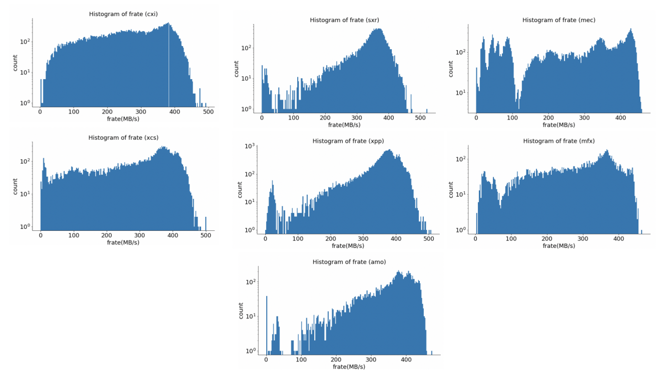

Figure 8: Distribution of Transfer Rate on Different Instruments: cxi and xcs have similar distribution; xpp, sxr and amo have similar distribution; mec and mfx have similar distribution. However, the three groups' distributions vary. The count is in logarithm scale.

From the data overview in Table 2, we also notice the close relationship between instrument and FFB file system, which is the source file system in the ANA transfer process. Therefore, we expect this source file system also affects the transfer rate, and we show the distribution plots of the two different source file systems in Figure 9.

3 MODELING FFB TRANSFERS

General Methodology. The raw data from the monitoring system at LCLS include a number of features that are important for the performance of the data transfers. These include: file size, instrument, target host, target file system, and node. The last four features are categorical features and are encoded using a one-hot encoding.

After a lengthy exploration of the data and prediction methods, we decided to add the following features to our prediction model:

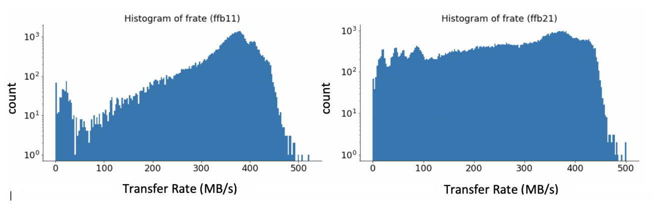

Figure 9: Distribution of Transfer Rate of Source File systems: As expected, the shapes of transfer rate from two sources are different. However, the shapes of the distributions mimic the two groups of distributions in Figure 8. This confirms the relationship between instruments and source file system.

- Statistics (transfer rate, file size, time between last job's stop time and current job's start time) from the last job on the same instrument, target file system, target host, and node, respectively;
- Statistics (same as above) from the most recently finished job of the same experiment chunk, if available;
- Time difference between the start time of the first job of the chunk and the start time of the current job;
- Number of total jobs and number of unique experiments running on the same trgfs, trghost, and node, respectively, when the current job is started;
- Time of day and day of week.

We call features of the first three types correlation features since they attempt to capture the correlation between jobs at different times. We call features of the fourth type concurrency features since they describe the concurrency on the system at a given time.

Due to the presence of categorical features, we opt to use decision tree-based methods in our regression model. There are several reasons that we believe tree based model will perform well. First, there are many categorical variables in the feature set. Tree based model can handle categorical variables, even though the computational complexity is high. Second, the relationship between the selected features and transfer rate is complex, so tree based model can learn the nonlinear relationship well. Finally, the tree model gives the feature importance of each feature, so it can provide clear interpretations of how each feature affects transfer rate. In particular, we use the random forest [3] and Xgboost models [1] on the features above to predict the transfer rate. For the random forest model, we omit the second new feature since the model cannot handle missing data. We tune the hyperparameters using nested cross validation (CV) for time series, as described in Algorithm 1. Nested CV mitigates information leaking from the training set to the validation set by having each element in the training set precede each element in the validation set during each iteration. This type of cross validation is important since we are using lagged features to predict future transfer rates. We measure the performance based on Root Mean Square Error (RMSE) in this process.

Algorithm 1 Nested Cross Validation

```
 1: Inputs: Training set ($X$), number of hyperparameters to try
    ($num\_params$), number of CV iterations ($k$), CV training and
    test widths ($train\_width$, $test\_width$), CV training and test set
    sizes ($train\_size$, $test\_size$)
 2: Sort $X$ in increasing order of start time
 3: for $i = 1$ to $num\_params$ do
 4:     Set $\alpha \leftarrow$ randomly sampled hyperparameters
 5:     for $j = 1$ to $k$ do
 6:         Set $train\_region \leftarrow$ random consecutive $train\_width$
            rows of $X$
 7:         Set $train\_set \leftarrow$ random subset of $train\_region$ of size
            $train\_size$
 8:         Set $test\_region \leftarrow$ $test\_width$ rows of $X$ following
            $train\_region$
 9:         Set $test\_set \leftarrow$ random subset of $test\_region$ of size
            $test\_size$
10:         Train Xgboost model on $train\_region$ and evaluate performance (RMSE) on $test\_region$, where the prediction is
            max(0, Xgboost prediction).
11:     end for
12:     if average of test RMSE's for $\alpha$ is lowest so far then
13:         Set $\alpha_{best} \leftarrow \alpha$
14:     end if
15: end for
16: return $\alpha_{best}$
```

Random Forest Results. Using the nested CV algorithm with the training set X equal to the first 90% of the rows, $k = 10$, $train\_width = 20000$, $train\_size = 5000$, $test\_width = 2000$, $test\_size = 500$, we then retrain the random forest model using the best hyperparameters on a 30000 row subset of the first 90% of the rows of X, and evaluate RMSE on a 3000 row subset of the last 10% of the rows of X, where the prediction is max(0, prediction). This yields an RMSE of 52.8MB/s. The following is the importance matrix for the ten most important features:

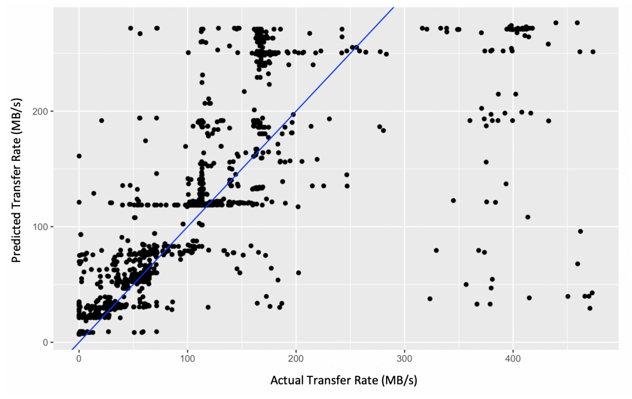

Figure 10: Actual vs Predicted plot using Random Forest

Feature	% Gain
Last Job Instrument, Transfer Rate	70.4%
Last Job File System, Transfer Rate	7.9%
Last Job Node, Transfer Rate	7.0%
File Size	4.1%
Last Job Instrument, Stop Time Difference	4.1%

We also plot the actual vs predicted transfer rate on the test set in Figure 10.

Xgboost Results. Using the nested CV algorithm with the training set X equal to the first 90% of the rows, $k = 10$, $train\_width = 20000$, $train\_size = 6000$, $test\_width = 2000$, $test\_size = 800$, we then retrain the Xgboost model using the best hyperparameters on the same 30000 row subset as the random forest, and evaluate the RMSE on the same 3000 row subset as the random forest, where again the prediction is max(0, Xgboost prediction). This yields an RMSE of 36.1MB/s. The following is the importance matrix for the ten most important features:

Feature	% Gain
Instrument	12.9%
Last Job of Same Experiment - File Size	9.5%
Day of the Week	7.9%
Last Job of Same Experiment - Start/Stop Time Diff.	7.5%
Target Host	6.2%
Last Job on Same Target File System - Transfer Rate	5.3%
Last Job on Same Node - Start/Stop Time Diff.	4.4%
Last Job of Same Instrument - Transfer Rate	3.9%

By comparing Figures 10 and 11, we can clearly observe the better performance of Xgboost. The points in Xgboost are more aligned with the y = x line. However, random forest displays right angle patterns along the y = x line, which indicates a poorer prediction result. In Figures 11 and 12 we show a few different plots of the errors on the test set before the New Year. We can see in Figure 12 that in general, our predictions are worse when the transfer rate is higher, particularly for instrument 'cxi.'

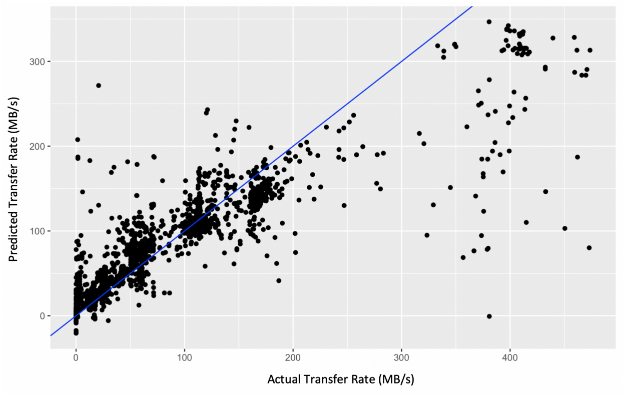

Figure 11: Actual vs Predicted plot using Xgboost

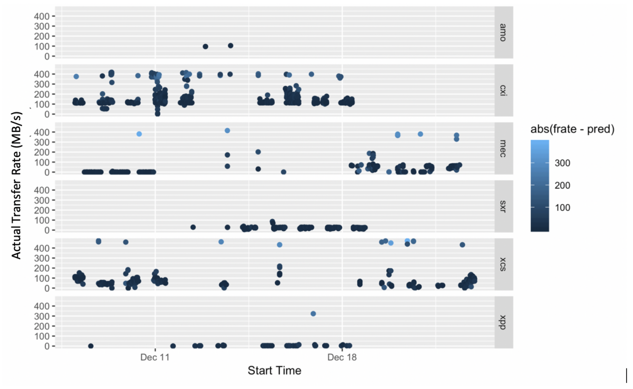

Figure 12: Transfer rates over time split by instrument with color corresponding to absolute prediction error and point size corresponding to file size.

4 MODELING ANA TRANSFERS

Time Independent Model. To predict transfer rate in ANA process, building an appropriate feature set is the key step. The first set of features is directly from the existing fields in the given data set. We perform basic statistical analysis and distribution analysis on the ANA file transfer rate based on all features available in the original data set. Out of the available features, we find that **instrument**, **source file system** and **target file host** are affecting the distribution of transfer rate significantly. The basic distribution plots are analyzed in section 2.

From the original data set, we also extract the information of the **experiment number** each file transfer belongs to. Files with the same experiment number belong together, and thus we would expect similar transfer rates for these files.

In the time independent model, the feature set is as follows. Among these features, instrument, srcfs and trgfs are encoded using one-hot encoding method, and experiment number is encoded using label encoding method.

- File size
- Instrument
- srcfs

Feature	Importance
File Size	66.827%
experiment number	17.325%
cxi(instrument)	6.153%
ffb11(srcfs)	4.855%
mfx(instrument)	2.604%

Table 5: Feature Importance of Time Independent Model

- trgfs
- Experiment number

With this feature set, we also choose decision tree based model to predict the file transfer rate. We implement gradient boosting [2] tree by splitting the data set into 90% training set and 10% testing set. The time independent model gives RMSE of **64.3MB/s**. From the feature importance matrix in Table 5 which only displays the 5 most important features, we notice that **file size** is the dominant factor that affects the transfer rate, and **experiment number** also has large importance on the transfer rate as we expected. The **instruments** count for approximately 10%, which also indicates its importance in predicting transfer rate. Specifically, **cxi** is the dominant one. This may be related to the fact that cxi is used most frequently to collect data, as shown in Table 1. We also notice that the top two instrument variables **cxi** and **mfx** represent the two different groups of distribution shapes, shown in Figure 8.

Time Dependent Model. In the previous model, we do not consider the relationship between consecutive file transfers and treat each file transfer independently. However, due to the time series nature of the data set, we believe that time effect and the relationship between consecutive file transfers can be important features to predict transfer rate. Therefore, we further improve the previous model by generating **lag variables** on transfer rate and file size. Five different types of lag variables are generated. Four of them depend on instrument, experiment, target file system and source file system. In other words, these lag variable are generated by taking the value from the most recently finished job on the same instrument, experiment, target file system or source file system. The fifth one takes the value from the most recently finished job regardless of the other four factors. There are some correlations between these lag variables. For example, transfer rate from the most recently finished job on the same instrument is highly likely to be the same as that in the same experiment, so we run experiments with different combinations of these five types of lag variables and choose the ones that give the best prediction result.

We are not only interested in lag 1, but also lags beyond the first one. By analyzing the auto-correlation between 20 lags and transfer rate, we notice that **lag 1** to **lag 5** have high correlation with transfer rate and **lag 5** is the inflection point among 20 lags. One sample auto-correlation plot in terms of source file system is shown in Figure 13.

In summary, in addition to the features used in the time independent model, in the time dependent model, we include the combination of lag variables. The complete feature set is shown below.

- File size
- Instrument

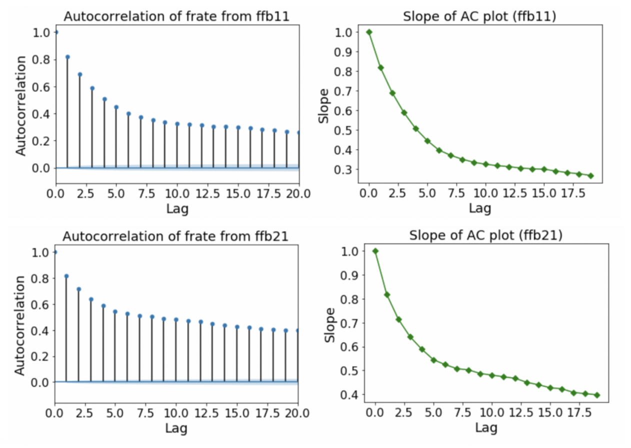

Figure 13: Auto-Correlation (AC) Plot of Transfer Rate on Source File System: The top two figures present the auto-correlation of transfers from source ffb11 and the bottom two figures present the auto-correlation of transfers from source ffb21. The two sets of figures show consistent patterns as described in the paragraph.

- srcfs
- trgfs
- Experiment number
- lag1 of file transfer rate and size on the same instrument, experiment, and overall
- lag5 of file transfer rate

Similar to the implementation in time independent model, we run Gradient Boosting Tree by splitting the data set into 90% training set and 10% testing set. We also run Random Forest using the same feature set so that we can compare the feature importance and the prediction result from two different tree models. Gradient boosting Tree gives RMSE **58.6 MB/s** and Random Forest gives RMSE **60 MB/s**. We make some observations from these prediction results. First, with the lag variables, we improve the prediction by **8.8%** in terms of RMSE. Second, Gradient Boosting Tree slightly outperforms in predicting the transfer rate.

By comparing Table 5 and Table 6, we can see the importance of file size decreases in time dependent model and the two lag1 variables become one of the most important features to predict transfer rate. Lag variables is indeed contributing to transfer rate prediction.

Table 6 and 7 show the importance feature matrix of the two tree models with lag variables included. From these 10 most important features we see that the two tree models both recognize lag variables as very important features. In Random Forest, lag variables have even higher feature importance than those in Gradient Boosting Tree, especially lag 5 becomes the third most important feature. This confirms that the state of network plays an important role in predicting transfer rate, no matter what model is implemented.

Feature	Importance
File Size	51.076%
lag1 on same instrument - transfer rate	14.064%
lag1 from same experiment - transfer rate	11.259%
experiment number	7.722%
cxi(instrument)	4.416%
lag5 overall - transfer rate	4.373%
lag1 overall - file size	3.274%
ffb11(srcfs)	1.437%
mfx(instrument)	0.802%
mec(instrument)	0.535%

Table 6: Feature Importance of GBM in Time Dependent Model

Feature	Importance
File Size	49.046%
lag1 from same experiment - transfer rate	15.443%
lag5 overall - transfer rate	8.969%
experiment number	6.945%
lag1 overall - file size	6.802%
lag1 on same instrument - transfer rate	6.091%
cxi(instrument)	3.817%
ffb11(srcfs)	0.744%
mfx(instrument)	0.461%
ana11(trgfs)	0.418%

Table 7: Feature Importance of Random Forest in Time Dependent Model

Based on the prediction result, we perform an error analysis on the predictions. The largest prediction error is more than **300MB/s**. A pattern shown for the records with large prediction errors is that the transfer rate of the preceding files with almost identical configurations are all at the same scale, but the transfer rate experiences a sudden increase or decrease for the current file, which has almost the same configuration as the previous ones. This sudden change is not captured by the models, and generates large prediction errors. One way we try to reduce this error is to include **time difference between the two consecutive file transfers** as one feature in the time dependent model. The reason is that if the two consecutive files are separated with a relative long time gap, the previous transfer rate should have smaller effects on the current one, so adding **time difference** feature should help. It turns out that, this feature leads to RMSE of **57.9MB/s** for Gradient Boosting Tree (1.2% decrease in RMSE), and **58.4MB/s** for Random Forest (2.6% decrease in RMSE).

Finally we tune the Gradient Boosting Tree using cross validation. By using the parameters: learning rate = 0.1, n estimators = 600, max features = 4.12, max depth = 11, min samples split = 700, min samples leaf = 10, we get RMSE **56.9MB/s**. The new feature importance for the tuned Gradient Boosting Tree with the feature **time difference between the two consecutive file transfers** added is shown in Table 8. We can see from the table that after adding **time difference** feature and applying cross validation, the importance score are more evenly distributed. Interestingly, importance of file size decreased by almost half, and the lag variables become the dominant factors. This is different from what we expected that **time difference** may decrease the transfer rate's dependence on

Feature	**Importance**
File Size	26.868%
lag1 from same experiment - transfer rate	22.614%
lag1 on same instrument - transfer rate	13.642%
lag1 overall - file size	9.254%
lag5 overall - transfer rate	8.766%
experiment number	6.496%
ffb11(srcfs)	3.599%
time difference between same experiment	2.940%
cxi(instrument)	1.937%
mec(instrument)	0.830%

Table 8: Feature Importance of GBM with Time Difference Feature and Cross Validation

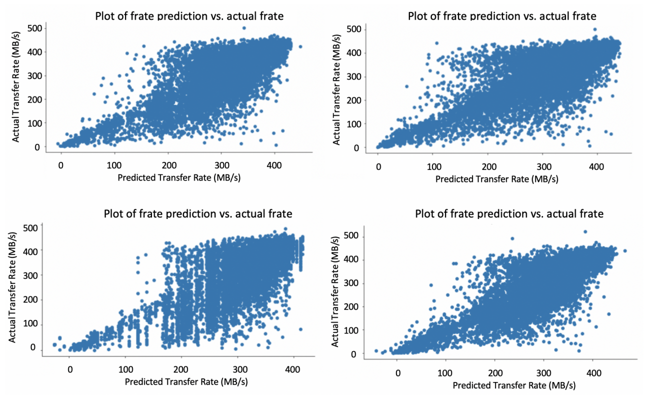

Figure 14: Predicted transfer rate vs. Actual transfer rate (Top Left: GBM in Time Dependent Model, Top Right: Random Forest in Time Dependent Model, Bottom Left: GBM in Time Independent Model, Bottom Right: GBM in Time Dependent Model with CV)

lag variables, and this may indicate that **time difference** is not the key to solve the problems for causing the large errors. Therefore, the issue discovered from error analysis will be one of the main things we plan to focus on next quarter. We will further study the patterns of the records with large prediction errors and incorporate the findings to improve the model.

Finally we present the plots of the actual transfer rate versus the predicted values from all the models mentioned above in Figure 14.

The plots also show that Gradient Boosting Tree slightly outperforms the Random Forest, Time Dependent Model outperforms Time Independent Model, and Time Dependent Model with cross validation generates the best prediction result as there are few points in the upper left and bottom right corners and points are more tightly aligned with y = x line in the fourth plot.

5 SUMMARY

Through our Exploratory Data Analysis (EDA) and model-building, we found that information about the status of the current system as well as statistics related to recent transfers allowed us to better predict the transfer rate of future transfers in both FFB and ANA process. Moreover, we discovered that seasonality plays a role in predicting transfer rates in FFB process. Seasonality remains as an area for future exploration. One other possible extension of this work would be to see if we can better describe when transfers are missing. For instance, in the asynchronous examples, we observed that transfers from certain hosts were delayed. It might be fruitful to develop a mechanism for identifying when we expect a transfer through different parts of the transfer process. In addition to this, it might be helpful to train the model on a larger, more recent data set. In our training process, we sampled 30,000 data points from the entire 9 months worth of data. It may be better to only train on the most recent 30,000 data points instead since we found out that the relationships between what parts of the system (e.g., nodes) are active can change as time passes, thereby making past relationships less relevant.

In terms of the performance of the tree-based models, Gradient Boosting Tree performs slightly better than Random Forest with the same feature set in both FFB and ANA process. The reason could be that Random Forest emphasizes more on lag variables in the model. As we described in the error analysis, lag variables are helping the prediction of most of the file transfers, but are also causing problems when the network changes suddenly and the transfer rate experiences a dramatic change. Exploring the features and possibly collecting more data that could explain the sudden transfer rate changes will be one of the main focuses of future work. In addition to tree based models, there are some other machine learning models such as Neural Network and SVM are also worth to try to predict the transfer rate in the future.

ACKNOWLEDGMENT

The authors would like to thank Kari Hanson and Professor Mary Wootters of Stanford University for setting up and sponsoring this project as part of ICME's Xplore project. This work was supported by the Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. Use of the Linac Coherent Light Source (LCLS), SLAC National Accelerator Laboratory, is supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under Contract No. DE-AC02-76SF00515.

REFERENCES

[1] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. *CoRR* abs/1603.02754 (2016). http://arxiv.org/abs/1603.02754
[2] Jerome H Friedman. 2001. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics* 29, 5 (2001), 1189–1232.
[3] Andy Liaw and Matthew Wiener. 2002. Classification and Regression by randomForest. *R News* 2, 3 (2002), 18–22. https://CRAN.R-project.org/doc/Rnews/
[4] Mengying Yang, Xinyu Liu, Wilko Kroeger, Alex Sim, and Kesheng Wu. 2018. Identifying Anomalous File Transfer Events in LCLS Workflow. In *AI-ScienceâĂŹ18*. ACM.