Learning Event Count Models with Application to Affiliation Ranking

Tam T. Nguyen
Laboratory for Systems, Software and Semantics
87 Gerrard St East
Toronto, ON M5B 1G9, Canada
nhanhtam@gmail.com

Ebrahim Bagheri
Laboratory for Systems, Software and Semantics
87 Gerrard St East
Toronto, ON M5B 1G9, Canada
ebrahim.bagheri@gmail.com

ABSTRACT
Event count prediction is a class of problems in time series analysis, which has been extensively studied over the years. Its applications range from the prediction of the number of publications in the scientific community to ATM cash withdrawal transaction prediction in the banking industry. However, in applied data science problems, using event count prediction models for real-world data often faces difficulties because the data violates not only the Poisson distribution assumption, i.e., the rate at which events occur should be constant, but the data is also relatively sparse, i.e., only a few event count values are greater than zero. Traditional techniques do not work well under these two conditions. To overcome these limitations, some researchers have proposed the generic autoregressive (AR) models for event count prediction, which work with non-constant event occurrence rates. As AR models solely use historical event count for forecasting, they might not be as flexible for incorporating domain knowledge. Moreover, and similarly, AR models may not work very well with the relatively short length-time series. In order to overcome these challenges, we propose a machine learning approach to address the event count prediction problem. We benchmark our proposed solution on the KDD Cup 2016 dataset by formalizing affiliation ranking as an event count time series prediction problem. We map the time series onto a highly dimensional state space and systematically apply the state-of-the-art machine learning algorithms to predict event counts. We then compare our proposed approach against solutions in the KDD Cup 2016 competition and show that our work outperforms the best models in this with an NDCG@20 score of 0.7573.

ACM Reference format:
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION
There are many approaches that deal with the problem of predicting event count data [2, 5, 11]. In [3], Brandt et al. have reviewed many time series approaches for event count data ranging from simple techniques such as autoregressive order $p$ (AR($p$)) [1] to more complex ones such as Poisson Exponentially Weighted Moving Average (PEWMA). Depending on the properties of the input data, one might choose a suitable approach to solve such a problem. A traditional approach for working with event count data has been to use AR models such as Gaussian ARIMA model [3]. This approach ignores how the count data is generated, which normally follows a certain statistical distribution. To overcome this issue, Poisson AR models [1] have been proposed. However, in practice, real world event count data do not always meet the Poisson distribution constraints. An alternative approach for tackling the problem is to use generalized event count models [12, 13].

Although event count prediction is a popular problem in political analysis [12], to the best of our knowledge, there is still a lack of well defined solutions for other applied data science problems, e.g., number of cash withdrawal transactions at ATMs, affiliation ranking based on the number of published papers, just to name a few. Given the affiliation ranking problem in KDD Cup 2016 competition as an instance of the event count prediction problem, the objective is to predict the number of publications or rankings of affiliations given the past history of publications. More specifically in the 2016 competition, the Microsoft Academic Graph (MAG) dataset from 2011 to 2015 [7, 17, 20] was given to the participants who were challenged to predict the values for 2016. Clearly, this use case is an instance of the event count time series forecasting problem. However, for each affiliation, there were only a few points that makes the time series analysis extremely difficult.

As such the problem we are trying to solve in this paper is a special case of the event count prediction problem [2, 5, 11] that has two main challenges:

- The available time series is rather short, i.e., we only have five data points for 2011 to 2015 for each event type, and therefore typical predictive models such as AR approaches do not work well on it.

- There are exogenous variables that impact the behavior of the time series but are unknown to the expert who is building the event count prediction model. For example, an author changing affiliation is also a cause for the change in affiliation ranking.

We propose that the general class of problems dealing with event count prediction with the above two major challenges can be addressed by using time series models, which can be mapped onto a highly dimensional state space [8]. On the basis of such mapping, we will show how state-of-the-art machine learning algorithms can be systematically applied to build predictive models. Doing
so, we are able to incorporate domain knowledge by embedding
domain-specific features to improve the performance of our predic-
tive models. The contributions of the paper are as follows:

- We show that a class of event count prediction problems
can be systematically addressed using machine learning

techniques.
- We discuss that affiliation ranking is an instance of this
class of problems.
- We detail our specific approach for affiliation ranking pre-
diction and report our experience of participating in the
KDD Cup 2016 competition and benchmark our solution
against the best submissions.

It is important to note that while for the sake of presenting
the details in this paper, the running theme of the paper is the
affiliation ranking problem, but the same systematic approach can
be generalized for similar event count prediction problems.

2 PROBLEM STATEMENT

In this section, we describe a generic event count forecasting problem
as well as its application to affiliation ranking prediction.

2.1 Event Count Prediction Description

The event count prediction problem can be described as shown in
Figure 1, where we have to predict the number of events happening
in the future given a set of events observed in the past. In general,
we are given an event dataset where each event may belong to
one or more classes and we would like to predict the number of a
certain class of events in the future. In order to address this problem,
we propose a systematic set of tasks: (i) aggregating event data to
generate event count time series; (ii) extracting features from the
time series; (iii) labeling the data within the training phase; (iv)
learning ensemble machine learning models; (v) predicting future
event counts. The details of these techniques will be presented in
the following sections within the context of the affiliation ranking
prediction problem.

2.2 Affiliation Ranking Description

The core dataset that was provided in the KDD Cup 2016 chal-
lenge was a snapshot of the Microsoft Academic Graph (MAG) [17]
dataset, which is freely available and includes information about
academic publications and citations. Being a heterogeneous graph,
MAG can be used to study the influential nodes of various types, in-
cluding authors, affiliations, and venues. The training data included
774 affiliations and 8,335 authors. On average, there are about 4.26
authors per paper.

In KDD Cup 2016 competition, the participants were challenged
to predict the affiliation rank in eight conferences including SIGIR,
SIGMOD, SIGCOMM, KDD, FSE, ICML, MM, and MOBICOM in
2016 given the publication data from 2011 to 2015. It consisted of 3
phases where each phase included 2 to 3 conferences. Before the
notification deadline of selected conferences, participants had to
submit their ranking predictions. The affiliation rankings would be
determined by the number of published papers belonging to that
affiliation in the conferences. To address the above event count
prediction, we treat a published paper as an event and an affiliation
as an event group. Affiliation ranking prediction then becomes forecasting the number of published papers of each affiliation in
2016.

3 MACHINE LEARNING APPROACH TO
EVENT COUNT PREDICTION

In this section, we present our proposed machine learning approach
for event count prediction.

3.1 Aggregation

The purpose of the aggregation phase is to convert the raw data into
event count time series, which can take many different forms. For
instance, one may be interested in both event counts and the ranks
of the counts. Moreover, one may be interested in aggregation of
grouped events instead of individual ones.

Taking the affiliation ranking problem as an example, we frame it based on the event count prediction problem. Let us treat the
publication of a paper as an event, then all co-authors of the paper
would have an event count of 1. If one is interested in the number of papers of an author, she may aggregate the event counts grouped by
author. Similarly, the event count of an affiliation can be calculated by aggregating the number of papers of all authors who are working for that affiliation.

Formally, let $A$, $P$, and $C$ be a set of all authors, papers, and
conferences, respectively. We define each paper to be represented by a selected set of its authors $p \in P$ and $p = \{a | a \in A\}$ and a
conference to be represented by a selected set of papers $c \in C$ and $c = \{p | p \in P\}$. With this formulation, we define author and affiliation scores as follows:

\[u(a, p) = \frac{s}{|p|}\]

(1)

The author score specifies that each co-author of a paper shares the same score and hence the score is disproportionate to the num-
ber of authors. For the sake of simplicity, one can choose $s = 1.0$.

As each author may have worked for different affiliations, we
define an affiliation score of a paper as follows:

\[v(o, p) = \frac{\sum_{a \in p} u(a, p)}{|\{(a, o) | a \in p\}|}\]

(2)

the number of affiliations that an author $a$ is working for

It is trivial to notice that the affiliation-paper score of a paper will be equal to the author score if she is the only one working at this affiliation. Otherwise, each affiliation will receive an equally distributed proportion.

\[x(o, c) = \sum_{p \in c} v(o, p)\]

(3)
We have four datasets as shown in Table 1 where we would like to predict. It depends on the event type as well as the where the three biggest diamonds represent Microsoft, University (rectangle) and its node size is proportional to its node score defined by assigning the coordinates as follows:

Figure 2 shows the KDD conference subgraph of the MAG dataset where each node represents an author (disc), an affiliation (diamond), or a paper (rectangle) and its node size is proportional to its node score defined in Equation 3. It is easy to see that there are a few affiliations that have many papers (more than 50 papers) in the KDD conference where the three biggest diamonds represent Microsoft, University of Illinois at Urbana Champaign, and Carnegie Mellon University.

### 3.2 Data Annotation

For the data annotation phase, we have to specify the target that we would like to predict. It depends on the event type as well as the time window that are used to aggregate the counts. For example in our problem, we would like to predict the annual affiliation score in the MAG dataset.

Formally, we calculate the historical affiliation scores \( x_t(o,c) \) of all affiliation and conference pairs for year \( t \in [2011, 2015] \). We have four datasets as shown in Table 1 where \( h \) is the feature function which will be discussed in the next subsection.

### 3.3 Feature Extraction

Before extracting features from the data, we perform aggregation to construct the time series of both annually scores and ranks of affiliations. We then apply a mapping function \( h \) to map the time series onto a state space [8]. Specifically, we create a state function \( h \) by assigning the coordinates as follows:

\[
\begin{align*}
h_1(t) &= y_t, \\
h_2(t) &= y_{t-p}, \\
\vdots \\
h_d(t) &= y_{t-(d-1)p}, \\
h_{d+1}(t) &= x_t, \\
h_{d+2}(t) &= x_{t-p}, \\
\vdots \\
h_{2d}(t) &= x_{t-(d-1)p}
\end{align*}
\]

where \( p \) is the delay time and \( d \) is the number of lag features or sub-dimensions of the state space. We refer to \( h(t) \) as feature vector at time \( t \). Given \( h(t) \) is directly generated from the time series, this process is usually referred to as autoregressive (AR) modeling. We adapt this approach by adding aggregation and entity profiling information and use machine learning algorithms. In general, one can derive at least four types of features as follows:

- **Lag score features**: It is reasonable to assume that a future score is highly correlated to its historical score. Thus, we use historical score values as a feature to predict the future score.
- **Lag rank features**: We assume that a future rank is highly correlated to its past rank. Thus, we use historical rank as a feature to predict the future rank.
- **Score aggregation features**: It is possible to hypothesize that the significant statistics of scores in the past represent the norm of the affiliation over time.
- **Entity profiling features**: It can be reasoned that entity profiles such as authors, conferences, and affiliations highly contribute to the increase in predictive power; therefore, these information can be incorporated into the model.

Table 2 shows the whole feature set. We use a prefix such as score, rank, and mean as well as a suffix \( t \) or \( t−d \) to discriminate between the features. For example, score_2014 is the score feature in 2014.

### 3.4 Modeling

Once we have mapped the time series into the state/feature space, the next step is to fit the models to the data. In this specific problem and according to the above four types of features, we have built a total of 106 features which consists of 4 score features, 4 rank features, and 98 count aggregation and profiling features. These features are used to build base models (including both baseline and individual) as presented in the following subsections.

#### 3.4.1 Baseline Models

In this section, we introduce two baseline approaches for event count prediction. They are generic autoregressive methods, which rely solely on lag features and do not depend on the application domain.
Table 1: Training and testing data.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation Train</td>
<td>$((h {c_i}<em>{i=2011..2013}, {p_i}</em>{i=2011..2013}, o, c), x_{2014}(o, c))</td>
</tr>
<tr>
<td>Validation Test</td>
<td>$((h {c_i}<em>{i=2012..2014}, {p_i}</em>{i=2012..2014}, o, c), x_{2015}(o, c))</td>
</tr>
<tr>
<td>Train</td>
<td>$((h {c_i}<em>{i=2011..2014}, {p_i}</em>{i=2011..2014}, o, c), x_{2015}(o, c))</td>
</tr>
<tr>
<td>Test</td>
<td>$((h {c_i}<em>{i=2012..2015}, {p_i}</em>{i=2012..2015}, o, c), {?})</td>
</tr>
</tbody>
</table>

Table 2: Feature set.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Formula</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>score_</td>
<td>$x_t(o, c)$</td>
<td>Lag score</td>
<td>Historical affiliation score features where $t \in [2011, 2014]$</td>
</tr>
<tr>
<td>rank_</td>
<td>$y_t(o, c)$</td>
<td>Lag rank</td>
<td>Historical affiliation rank features where $t \in [2011, 2014]$</td>
</tr>
<tr>
<td>[agg]_</td>
<td>$\text{agg}(x_t(o, c))$</td>
<td>Score aggregation</td>
<td>Score aggregation features where agg can be min, max, std</td>
</tr>
<tr>
<td>[agg]_</td>
<td>$\text{agg}(x_t(o, c))$</td>
<td>Score aggregation</td>
<td>Score aggregation features where agg can be mean, median, count</td>
</tr>
<tr>
<td>[agg]<em>aff_author</em></td>
<td>$\text{agg}(x_t(o, a))$</td>
<td>Entity profiling</td>
<td>Affiliation-author profiling features where agg can be min, max, std, mean, and median</td>
</tr>
<tr>
<td>[agg]<em>aff_paper</em></td>
<td>$\text{agg}(x_t(o, p))$</td>
<td>Entity profiling</td>
<td>Affiliation-paper profiling features where agg can be min, max, std, mean, and median</td>
</tr>
<tr>
<td>[agg]<em>aff_paper</em></td>
<td>$\text{agg}(x_t(o, c))$</td>
<td>Entity profiling</td>
<td>Affiliation-conference profiling features where agg can be min, max, std, mean, and median</td>
</tr>
</tbody>
</table>

- **Average Affiliation Scores (AAS) or rolling mean**: We calculate the simple average of affiliation scores.
- **Weighted Average Affiliation Scores (WAAS) or linear regression**: Similar to AAS except that recent affiliation scores are weighted higher. We use weights of 0.6, 0.7, 0.8, 0.9 and...

3.4.2 Individual (Single) Models. To study whether combining generic features with domain-specific features works well, it is possible to build regression models that use all of the features. In our case, we built six regression models. Among them, only profiling features are domain-specific features. The other features can be directly determined from the time series. The algorithms and their parameters can be chosen based on their performance on the validation dataset using a grid search as shown in Table 3.

3.4.3 Ensemble Method. To bring the performance of the predictive models to the next level, we propose the use of stacked generalization [21] to build ensemble predictions of multiple base models. Stacked generalization uses the predictions of base models, called level-0 generalizers, as inputs for another model, called level-1 generalizer, to generate ensemble predictions.

To maximize prediction performance of stacked generalization, it is possible to use the linear regression level-1 generalizer and greedy forward selection of base models. Ting and Witten have already shown that stacked generalization achieves the best accuracy when linear models without non-negativity constraints are used as the level-1 generalizer [19]. The greedy forward selection starts with the base model with the highest validation score. Then, at each step, it adds an additional base model to the ensemble. It trains a linear regression model with validation predictions of base models in the ensemble, and calculates its validation score. Once the validation score stops improving, the selection stops. Then, it uses the linear regression model to generate ensemble test predictions from the test predictions of the selected base models.

3.5 Validation

Normalized Discounted Cumulative Gain (NDCG)@20, which is popular in information retrieval [14], is used as an evaluation metric to measure relevance [10]:

\[ DCG_n = \sum_{i=1}^{n} \frac{rel_i}{\log_2(i+1)} \]  

\[ NDCG_n = \frac{DCG_n}{IDCG_n} \]

where \( i \) is the rank of an affiliation, and \( rel_i \) is its relevance score. IDCG is the ideal DCG score, or the maximum possible DCG score, and is calculated by sorting affiliations of a result list by relevance up to position \( n \).

As the training data is time dependent, the traditional validation techniques such as random train-test split and k-fold cross validation don’t work. Moreover, because the testing data is unseen, we have to separate the training data so that it is hidden when we generate features for the validation data. Otherwise, our models will be over-fitting to the training data and lack of generalization on the testing data. Therefore, we propose the out-of-time (OOT) validation to preserve the temporal relationship between the training and test data sets. Figure 3 shows the validation process which consists of two phases: training-testing and validation-testing.

For validation-testing, we use data until 2014 as input variables, and predicted affiliation rankings in 2015. Then, we calculated validation NDCG@20 scores of the models with their validation test predictions and affiliation rankings in 2015. We are based on these validation NDCG@20 scores to select and optimize the parameters of our models. After that, we apply our models to the training-testing phase.

In the training-testing phase, we use data until 2014 as input variables and affiliation rankings in 2015 as a target variable for training. For testing, we use data until 2015 as input variables, and predict affiliation rankings in 2016. Similar to the validation-testing phase, we use data until 2013 as input variables and affiliation rankings in 2014 as a target variable.

4 PERFORMANCE EVALUATION

In this section, we evaluate the performance of our models using Normalized Discounted Cumulative Gain (NDCG)@20 [14], benchmark them with simple baseline models, and compare our results to other teams in the KDD Cup 2016 competition. Moreover, we also study the importance of our feature set as well as the correlation of the top features.

4.1 Performance on Training Data

The performance of the base models vary across eight conferences. In the validation test with 2015 ground truth data, WAAS and GBM score the best for three and two conferences, respectively. RR, RF, and XGB score the best for one conference. Table 4 shows the validation NDCG@20 scores of base models for the eight conferences in all phases. In general, a generic event count prediction approach with a linear regression model outperforms the other models on two out of eight conferences. It means that embedding more domain-specific features improves the performance of the predictive models.

Table 5 shows the performance results of ensemble models using the stacked approach. We solely rely on cross validation results to choose the base models. By further investigating our ensemble models, we notice that model selection results as well as NDCG@20 scores vary across the eight conferences. It means that there would not be any global model that would work well on all conferences. For a few conferences, base models outperformed ensemble ones. Therefore, we combine the best model on each conference to make the final prediction. For example, we select RF, GBM, an ensemble model of \{RR, RF, XGB, LR\} for SIGIR, KDD, and MM conferences, respectively solely based on the validation data.

4.2 Top Features

In this section, we use Random Forests to study the importance of the features. We categorize our features into three groups: lag, aggregation, and profiling. Table 6 (left) shows top score lag features. All historical score features are ranked as top 4 features by the RF model. Especially, scores in both 2014 and 2013 are the most relevant to the target (score in 2015 as the target of the validation) of the model. This suggests that older historical data is less relevant to the future outcome.

Similarly, Table 6 (right) shows top lag rank features. Both rank and score of affiliations are ranked in the top 11 features. To understand why lag score features work better than rank features,
Table 3: Selected machine learning algorithms.

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LR</td>
<td>Linear Regression [15]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fit_intercept=True</td>
</tr>
<tr>
<td>2</td>
<td>RR</td>
<td>Ridge Regression [16]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>alpha=1.0</td>
</tr>
<tr>
<td>3</td>
<td>SVR</td>
<td>Support Vector Regressor [18]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C=1.0, degree=3, epsilon=0.1, gamma='auto', kernel='rbf'</td>
</tr>
<tr>
<td>4</td>
<td>RF</td>
<td>Random Forests [4]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_estimators=100, criterion='mse'</td>
</tr>
<tr>
<td>5</td>
<td>GBM</td>
<td>Gradient Boosting Machine [9]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>alpha=0.9, learning_rate=0.1, max_depth=3, n_estimators=100, subsample=1.0, loss='ls'</td>
</tr>
<tr>
<td>6</td>
<td>XGB</td>
<td>eXtreme Gradient Boosting [6]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>objective='reg:linear', max_depth=5, eta=0.01, subsample=0.8, colsample_bytree=0.4,</td>
</tr>
</tbody>
</table>

Figure 3: Validation process.

Table 4: Validation NDCG@20 scores of base models.

<table>
<thead>
<tr>
<th>Base Model</th>
<th>SIGIR</th>
<th>SIGMOD</th>
<th>SIGCOMM</th>
<th>KDD</th>
<th>ICML</th>
<th>MM</th>
<th>FSE</th>
<th>MOBICOM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAS</td>
<td>0.8054</td>
<td>0.8020</td>
<td>0.7903</td>
<td>0.7951</td>
<td>0.8924</td>
<td>0.5963</td>
<td>0.5785</td>
<td>0.6860</td>
</tr>
<tr>
<td>WAAS</td>
<td>0.8241</td>
<td>0.8107</td>
<td>0.7946</td>
<td>0.7792</td>
<td>0.8959</td>
<td>0.6085</td>
<td>0.6137</td>
<td>0.6998</td>
</tr>
<tr>
<td>LR</td>
<td>0.7091</td>
<td>0.4702</td>
<td>0.8924</td>
<td>0.6778</td>
<td>0.1564</td>
<td>0.6515</td>
<td>0.5475</td>
<td>0.5916</td>
</tr>
<tr>
<td>RR</td>
<td>0.7083</td>
<td>0.5272</td>
<td>0.8869</td>
<td>0.7726</td>
<td>0.8683</td>
<td>0.6529</td>
<td>0.5683</td>
<td>0.6652</td>
</tr>
<tr>
<td>SVR</td>
<td>0.5385</td>
<td>0.4494</td>
<td>0.4525</td>
<td>0.6865</td>
<td>0.7431</td>
<td>0.2861</td>
<td>0.1949</td>
<td>0.3765</td>
</tr>
<tr>
<td>RF</td>
<td>0.8667</td>
<td>0.5063</td>
<td>0.7299</td>
<td>0.6057</td>
<td>0.6360</td>
<td>0.6406</td>
<td>0.5894</td>
<td>0.3973</td>
</tr>
<tr>
<td>GBM</td>
<td>0.5812</td>
<td>0.5190</td>
<td>0.9205</td>
<td>0.8073</td>
<td>0.6036</td>
<td>0.6478</td>
<td>0.6324</td>
<td>0.6160</td>
</tr>
<tr>
<td>XGB</td>
<td>0.4590</td>
<td>0.4011</td>
<td>0.4001</td>
<td>0.8520</td>
<td>0.8130</td>
<td>0.6269</td>
<td>0.5980</td>
<td>0.6292</td>
</tr>
</tbody>
</table>

we further studied the correlation between them as shown in Figure 4. Let us start with the score_2014 column: it is easy to see that score_2014 is highly correlated with score_2013 with a coefficient of 0.734. The next one is score_2012 with a coefficient of 0.619. Finally, the correlation coefficient of score_2014 and score_2011 is 0.563. We also have identical results within the rank features. These results are consistent for all score and lag features. This explains why AAS and MAAS models tend to work very well for this dataset. Although rank features are less important than score features, they are ranked much higher than aggregation features. Table 7 shows top aggregation features. Only two features, the average of the number of authors per paper in an affiliation and the standard
deviation of the total number of authors, work better than rank features. In general, aggregation features have less importance than their score and rank feature counterparts. This is because score and rank features are directly relevant to the target as mentioned earlier.

### 4.3 Performance on Testing Data

In this section, we compare our test NDCG@20 score with top 3 teams in the final leaderboard. In Phase 1 (SIGIR conference), our solution outperforms all other teams where the improvement is up to 3%. In Phase 2 (KDD conference), our score is more than 0.77 which is stable compared with Phase 1. Finally, our model achieves the best score of 0.7345 in Phase 3 (MM conference). With these good results in the three phases, our overall score is 0.7573 which outperforms all other teams. This score is the best score on NDCG among all participating teams.

As seen from the reasonable performance results, we believe that the systematic approach for converting the event count problem into a time series model that would be used for deriving four types of features is a systematic and reliable approach for achieving good performance on problems of this nature.

### 5 CONCLUSION

In this paper, we presented our autoregressive solution for the general class of event count prediction. More specifically, we reported our successful experience with the 2016 KDD Cup dataset for finding highly influential affiliation rank prediction on the Microsoft Academic Graph dataset. We proposed a systematic and generalizable approach for the event count prediction problem by generating historical score and rank features based on a time series representation, which would then be used to train regression models to solve the problem. Our study shows that historical scores and ranks are strong indicators for future ranking. Moreover, our

---

Table 5: Selected base models and validation scores of ensemble models.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Conference</th>
<th>Selected Base Models</th>
<th>NDCG@20</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SIGIR</td>
<td>RF</td>
<td>0.8667</td>
</tr>
<tr>
<td></td>
<td>SIGMOD</td>
<td>WAAS</td>
<td>0.8107</td>
</tr>
<tr>
<td></td>
<td>SIGCOMM</td>
<td>GBM</td>
<td>0.9205</td>
</tr>
<tr>
<td>2</td>
<td>KDD</td>
<td>GBM</td>
<td>0.8520</td>
</tr>
<tr>
<td></td>
<td>ICML</td>
<td>AAS, WAAS</td>
<td>0.9193</td>
</tr>
<tr>
<td>3</td>
<td>MM</td>
<td>RR, RF, XGB, LR</td>
<td>0.6673</td>
</tr>
<tr>
<td></td>
<td>FSE</td>
<td>LR, XGB</td>
<td>0.7179</td>
</tr>
<tr>
<td></td>
<td>MOBICOMM</td>
<td>WAAS, GBM</td>
<td>0.7399</td>
</tr>
</tbody>
</table>

Table 6: Top score and rank features.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Name</th>
<th>Score</th>
<th>Rank</th>
<th>Name</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>score_2014</td>
<td>0.390674</td>
<td>6</td>
<td>rank_2014</td>
<td>0.021664</td>
</tr>
<tr>
<td>2</td>
<td>score_2013</td>
<td>0.144808</td>
<td>7</td>
<td>rank_2012</td>
<td>0.021036</td>
</tr>
<tr>
<td>3</td>
<td>score_2012</td>
<td>0.056851</td>
<td>8</td>
<td>rank_2011</td>
<td>0.020587</td>
</tr>
<tr>
<td>4</td>
<td>score_2011</td>
<td>0.042812</td>
<td>11</td>
<td>rank_2013</td>
<td>0.016911</td>
</tr>
</tbody>
</table>

---

The performance of each team can be found in the official KDD Cup 2016 website: https://kddcup2016.azurewebsites.net
experimental results show that applying model selection on each subset of the data would be a good choice for improving performance. We hope our solution would serve as a solid stepping stone for practitioners in event count prediction, which can be applied to various domains such as affiliation ranking, ATM transaction prediction, queue length prediction, and traffic jam prediction.

Table 7: Top aggregation and profiling features.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Name</th>
<th>Score</th>
<th>Rank</th>
<th>Name</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>mean_aff_num_authors_per_paper</td>
<td>0.022054</td>
<td>20</td>
<td>max_aff_author_cnt</td>
<td>0.007306</td>
</tr>
<tr>
<td>9</td>
<td>std_aff_author_cnt</td>
<td>0.020489</td>
<td>21</td>
<td>max_aff_paper_cnt</td>
<td>0.006524</td>
</tr>
<tr>
<td>10</td>
<td>std_aff_paper_cnt</td>
<td>0.018200</td>
<td>22</td>
<td>aff_paper_cnt_1</td>
<td>0.006509</td>
</tr>
<tr>
<td>12</td>
<td>mean_num_papers_per_conference</td>
<td>0.015603</td>
<td>23</td>
<td>mean_aff_author_cnt</td>
<td>0.006062</td>
</tr>
<tr>
<td>13</td>
<td>aff_author_cnt_2</td>
<td>0.011357</td>
<td>24</td>
<td>aff_author_cnt_0</td>
<td>0.005983</td>
</tr>
<tr>
<td>14</td>
<td>std_aff_conference_cnt</td>
<td>0.011271</td>
<td>25</td>
<td>median_aff_paper_cnt</td>
<td>0.005892</td>
</tr>
<tr>
<td>15</td>
<td>aff_paper_cnt_3</td>
<td>0.010608</td>
<td>26</td>
<td>aff_paper_cnt_2</td>
<td>0.005443</td>
</tr>
<tr>
<td>16</td>
<td>aff_author_cnt_3</td>
<td>0.009654</td>
<td>27</td>
<td>aff_conference_cnt</td>
<td>0.004746</td>
</tr>
<tr>
<td>17</td>
<td>median_aff_author_cnt</td>
<td>0.008396</td>
<td>28</td>
<td>aff_paper_cnt</td>
<td>0.004710</td>
</tr>
<tr>
<td>18</td>
<td>aff_author_cnt_1</td>
<td>0.008224</td>
<td>29</td>
<td>min_aff_author_cnt</td>
<td>0.004471</td>
</tr>
<tr>
<td>19</td>
<td>aff_author_cnt</td>
<td>0.008128</td>
<td>30</td>
<td>aff_paper_cnt_0</td>
<td>0.003957</td>
</tr>
</tbody>
</table>

Table 8: Final NDCG@20 scores and results.

<table>
<thead>
<tr>
<th>Team</th>
<th>Phase 1</th>
<th>Phase 2</th>
<th>Phase 3</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our approach</td>
<td>0.7734</td>
<td>0.7720</td>
<td>0.7345</td>
<td>0.7573</td>
</tr>
<tr>
<td>burebistas</td>
<td>0.7491</td>
<td>0.7970</td>
<td>0.7082</td>
<td>0.7519</td>
</tr>
<tr>
<td>ami@TASL</td>
<td>0.7364</td>
<td>0.7815</td>
<td>0.7278</td>
<td>0.7510</td>
</tr>
<tr>
<td>mmmlab_iitd</td>
<td>0.6721</td>
<td>0.8075</td>
<td>0.7265</td>
<td>0.7480</td>
</tr>
</tbody>
</table>

REFERENCES