Are Crowded Events Forecastable from Promotional Announcements with Large Language Models?

Soto Anno
Tokyo Institute of Technology
Tokyo, Japan
anno@miubiq.cs.titech.ac.jp

Dario Tenore
ETH Zurich
Zurich, Switzerland
dtenore@student.ethz.ch

Kota Tsubouchi
LY Corporation
Tokyo, Japan
ktsubouc@lycorp.co.jp

Masamichi Shimosaka
Tokyo Institute of Technology
Tokyo, Japan
simosaka@miubiq.cs.titech.ac.jp

ABSTRACT

Forecasting the number of visitors at a public event, termed *event crowd forecasting (ECF)*, has recently garnered attention due to its social significance. Although existing ECF methods have pioneered successful feature design by considering event contents with contexts (e.g., weather, type of day, time), their scalability across different event types is limited due to the necessity of costly feature engineering. To address this issue, we propose a novel ECF framework, named **EventOutlook**. Based on our observation of various events, online event announcements indicate the factors that induce crowded events. Thus, we incorporate event announcements into ECF methods. To handle such unstructured data, which have no unified format among events, we leverage large language models (LLM) to extract crowding factors and embed them into an *LLM-driven crowding-indicator feature (LCIF)*. Empirical experiments with real-world event data show that EventOutlook significantly improved ECF performance compared to state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → *Information systems applications*; • **Human-centered computing** → *Empirical studies in ubiquitous and mobile computing*;

KEYWORDS

Event Crowd, Event Attendance, Large Language Model, Crowd Forecasting, Mobility Logs, Event Announcement

ACM Reference Format:

Soto Anno, Dario Tenore, Kota Tsubouchi, and Masamichi Shimosaka. 2024. Are Crowded Events Forecastable from Promotional Announcements with Large Language Models?. In *The 32nd ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL '24), October 29-November 1, 2024, Atlanta, GA, USA*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3678717.3691240

SIGSPATIAL '24, October 29-November 1, 2024, Atlanta, GA, USA

ACM ISBN 979-8-4007-1107-7/24/10.
https://doi.org/10.1145/3678717.3691240

1 INTRODUCTION

What are typical factors and conditions that attract many visitors to events? In general, more people will go to an event if its contents are attractive, if the venue is easily accessible, and if the event occurs on a weekend or sunny day. The task of forecasting the number of visitors for public events, which we call *event crowd forecasting (ECF)*[1], plays a critical role in event marketing and demand management [5, 6]. In fact, several researchers have addressed event crowd forecasting with the recent advances in machine learning techniques [2, 12].

Existing methods have realized ECF by using the event's contents and the associated contextual information, such as the day of the week and weather for their input features [2, 12]. As an example, imagine a soccer game between two professional teams, where many supporters of each team will attend the game in the stadium. In that case, the crowd can be forecasted by using game-related statistics (e.g., the player's abilities and past match results) and the importance of the game (e.g., whether it is a cup final or an ordinary league match) as the event's contents for the input features [12].

Although existing methods are successful by taking the list of features as input, such a listing strategy limits the model's scalability to other types of events, as different events have different contents that attract visitors. For example, the event crowd in a concert may vary depending on the performer's popularity, while the number of participants in an academic conference may vary depending on the research field. Listing the event contents and engineering the features for all the different types of events requires expert knowledge [10], which is impractical in terms of costs.

Based on our observation of various events, event announcements published on the web by organizers indicate the factors that induce crowded events. For example, the fireworks festival held in Tokyo in 2019 was announced as follows: *"The Sumida River Fireworks Festival has a long history, ... This is an essential event in Tokyo's summer night sky... We are going to set off 20,000 fireworks, which is one of the largest scale in Japan."* As exemplified, such announcements generally include detailed descriptions of their contents (e.g., "20,000 fireworks") and promotional lines (e.g., "long history," "an essential event," "one of the largest scale," etc.) to advertise the attractiveness of events. Nowadays, many events, regardless

[1] In this paper, we define *event crowd forecasting (ECF)* as the task of forecasting the number of visitors to an event, in order to distinguish it from the *event attendance prediction* problem, which predicts whether or not an individual attends an event.

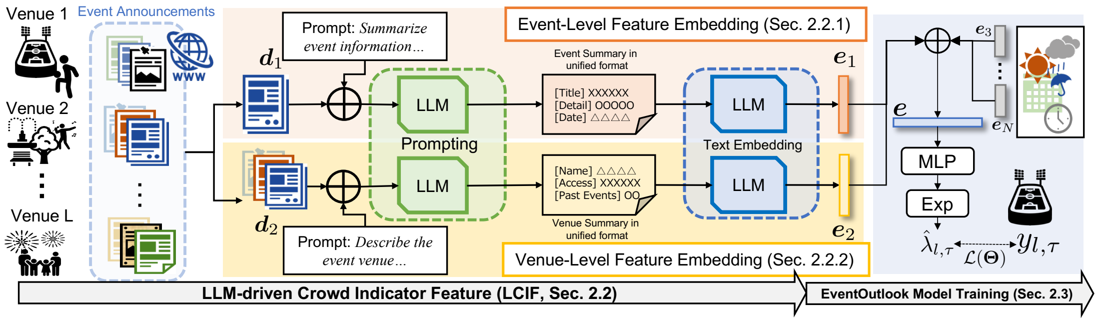

Figure 1: Overview of the EventOutlook framework. EventOutlook generates two types of LLM-driven features, i.e., an event-level feature and a venue-level feature from the event announcements. The input features, including the LLM-driven features and other features obtained from external contexts, are fed into the MLP to predict the event crowd at each venue.

of their event types, are announced in advance on the web. Thus, we assume that such announcements contain information useful for capturing future event crowds for various types of events.

However, extracting the event contents from such announcements is challenging because the data is unstructured, and there is no unified format among events. For instance, a sports game might only announce the opposing teams, but other events might give a detailed history to make the event more attractive. Thus, realizing the ECF for various types of events remains challenging.

To address this issue, we propose **EventOutlook** , a novel framework for forecasting crowds in various public events. To address the first challenge of unstructured data without a unified format, we focus on large language models (LLMs) that have recently succeeded in extracting high-quality representations from textual data. We leverage an LLM-based prompting and text embedding model to obtain an **LLM-driven crowding indicator feature (LCIF)** that reflects rich semantics about events and venues extracted from unstructured event data.

Our contributions can be summarized as follows: (1) we propose the EventOutlook framework to forecast event crowds for various types of events by modeling the effect of event contents on the event crowd by leveraging the event announcements available on the web; (2) we introduce an LLM-based crowding indicator feature (LCIF), which can be automatically created from unstructured event information; (3) we evaluate EventOutlook based on one year's event information, including various events held in 24 event venues, and the results show that the proposed framework outperforms state-of-the-art ECF methods.

2 PROPOSED EVENTOUTLOOK FRAMEWORK

The problem with existing event crowd forecasting methods is that their predictions rely on costly feature engineering for target event types [2, 12]. To address this issue, we propose the EventOutlook framework, whose schematic diagram is illustrated in Fig. 1. The key idea of EventOutlook is to leverage the event promotional announcements as the input features by utilizing LLMs.

2.1 Problem Formulation

The number of event visitors around the event venues is modeled as an event crowd. We denote $y_{l,\tau}$ as the number of people around the venue l at time segment τ. The time length of τ is set to an hour to deal with the forecasting in a fine-grained time resolution. The number of people in each time segment takes a non-negative integer, which is akin to count data [8]. Thus, it can be assumed that $y_{l,\tau}$ follows a Poisson distribution. Following a prior study [3, 4], we model the Poisson likelihood of $y_{l,\tau}$ written as $p(y_{l,\tau}) = \mathrm{Pois}(y_{l,\tau}; \lambda_{l,\tau}) = \lambda_{l,\tau}{}^{y_{l,\tau}} \exp(-\lambda_{l,\tau})/y_{l,\tau}!$, where $\lambda_{l,\tau} > 0$ denotes the mean parameter of the Poisson distribution. Now, our task turns into estimating the mean parameter $\lambda_{l,\tau}$.

To estimate $\lambda_{l,\tau}$, we can leverage input features such as event contents, venue information, day of the week, type of day (weekday or weekend), weather, and time. We denote these features as $\mathcal{D} = \{\boldsymbol{d}_1, \boldsymbol{d}_2, ..., \boldsymbol{d}_N\}$, where N is the number of features. We regard $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ as the event contents and venue information, respectively. We leverage an announcement text for each event to formalize $\boldsymbol{d}_1$, while we use announcement texts of multiple events held at the target venue to formalize $\boldsymbol{d}_2$. We regard $\boldsymbol{d}_3$, ..., $\boldsymbol{d}_N$ as the other features (e.g., weather).

However, $\lambda_{l,\tau}$ cannot be simply regressed by $\boldsymbol{d}_1, ..., \boldsymbol{d}_N$ because $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ are unstructured texts. While $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ are unstructured, $\boldsymbol{d}_3$, ..., $\boldsymbol{d}_N$ are typically formulated in a one-hot encoding format to represent the context (e.g., weather) and time. In such case, $\boldsymbol{d}_3$, ..., $\boldsymbol{d}_N$ are simply encoded into the learnable embedding vectors $\boldsymbol{e}_3$, ..., $\boldsymbol{e}_N$ by following common practice [9]. To solve the issue of handling unstructured texts of $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$, we generate the feature embeddings $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ (corresponding to $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$, respectively) by leveraging the LLMs.

2.2 LLM-driven Crowd-Indicating Feature

Fig. 2 shows raw event announcements crawled from the web. Our key observation is that such announcements include detailed descriptions, ranging from the event's contents to its schedules. People check the announcements and then decide whether or not to attend the event. Thus, we came up with the idea of leveraging the announcements to forecast the event crowd. However, the announcements involve unstructured texts, and the text format is not unified between, e.g., the fireworks festival shown in Fig. 2(a) and the soccer championship shown in Fig. 2(b).

Meanwhile, LLMs (e.g., GPT-4 [1]) have been emerging, and have achieved promising performance in text extraction by prompting and the generation of text embeddings [7]. Inspired by this, we realize the automatic feature creation based on event announcements

Sumida River Fireworks Festival (42nd)

Sumida River Fireworks Festival, a summer tradition in Tokyo. The event has been held since the Edo period and is the most famous fireworks festival in Japan.
...
This fireworks festival is an essential event in Tokyo's summer night sky.
...
Number of launched fireworks: 20000
Date & Time: Saturday, July 27, 2019, 19:00-20:30
Access: 15 min. walk from Asakusa Subway Station
...

(a) Fireworks Festival.

Emperor's Cup JFA 99th Japan Football Championship Final

Schedule Type
Event Name
Match Card

2020/1/1 (Wed, holiday) Soccer
Emperor's Cup JFA 99th All Japan Soccer Championship Final
Vissel Kobe vs Kashima Antlers

(b) Soccer Championship Final.

Figure 2: Examples of event announcements for two different events. They are unstructured data, and the format of the text is not unified between the two events.

by leveraging the LLM as follows: (1) We automatically extract the event contents and venue information in a unified format from the raw announcement texts by prompting the LLM. (2) We convert the extracted information into the embedding vectors by utilizing the LLM-based text embedding model.

2.2.1 Event-Level Feature Embedding. First, we create a feature vector for each event from the individual event announcements. We design a *prompt* to instruct the LLM to extract the following information: event title, overall description, event category, schedules, ticket price (if applicable), venue name, and access to the venue (if available). We concatenate the designed prompt and the raw announcement of an event into a single text, and then we feed it to the LLM. Finally, we apply the LLM-based text embedding model [1] to this prompting result to obtain an embedding vector. We describe the used prompt for the LLM in Section 3.

2.2.2 Venue-Level Feature Embedding. Another feature is a venue-level feature vector, which characterizes event venues. Although the venue information is partially targeted in the event-level feature, we have found that some event announcements only focus on the event contents and lack the venue information. Therefore, we extract the venue-level information from all the historical events held in the target venue. The extracted information regarding the venue comprises the overall description, usual events, accessibility, and crowding indicators such as the capacity of the venue (if available). Similar to the event-level feature creation, we prompt the LLM to extract the venue-level information, and then convert the resulting text into an embedding vector.

2.3 EventOutlook Model Training

$\lambda_{l,\tau}$ is regressed by the concatenation of the feature-wise embeddings, denoted as $\boldsymbol{e} = [\boldsymbol{e}_1^\top \cdots \boldsymbol{e}_N^\top]^\top \in \mathbb{R}^E$ by using an MLP. While any activation function can be employed for the hidden layer of the MLP, the final layer of the model is followed by an exponential function for ensuring $\lambda_{l,\tau} > 0$. Since we assume that the $y_{l,\tau}$ follows a Poisson distribution, the EventOutlook model can be trained by minimizing the negative log-likelihood of the Poisson distribution. This is formulated as $\mathcal{L}(\Theta) = -\sum_l \sum_\tau \ln \mathrm{Pois}(y_{l,\tau}; \lambda_{l,\tau})$, where Θ represents the learnable parameters of the model, such as the parameters both for the MLP used to embed the feature $\boldsymbol{d}_3, \ldots, \boldsymbol{d}_N$ and the MLP for regressing $\lambda_{l,\tau}$. The model can be trained in an end-to-end fashion.

3 EXPERIMENTS

3.1 Experimental Setup

3.1.1 Dataset. As the records for the number of event visitors, we used crowd flow data collected from individual location data sent from smartphones running an app that collects GPS transmissions. The GPS logs were collected between February 1st, 2019 and January 31st, 2020 through a mobile application from LY Corporation. Each record, collected with user consent, was entirely anonymized by replacing the user IDs with dummy identifiers and was characterized by timestamp, latitude, and longitude information. We aggregated the mobility logs within each event venue on an hourly base demarcated by a 500×500m square area, and tabulated their quantity as event crowds. Hence, we refrained from using any datasets with personally identifiable information for the data analysis and model construction. We also used event announcement data obtained from Infomotion, Inc[2]. Each event record was crawled from the web, containing a title, description with HTML tags, event category (e.g., sports, music, etc), schedules (starting/ending date and time), venue name, venue address, and URL reference. We obtained the data from 302 public events held in 24 event venues in Japan, such as large sports stadiums, concert halls, event sites, or riversides (where fireworks festivals were usually held), between February 1st, 2019, and January 31st, 2020.

```
Summarize event information given the original messy information extracted from web pages.

Requirements:
- INCLUDE ONLY information RELEVANT to event popularity.
- ALWAYS MAKE SURE to output only the facts.
- DO NOT include the information that cannot be known before the event is held (e.g., the scores of games).
- Use the present tense.
- Be clear, complete, and concise WITHIN 100 words.
- Identify specific event category (e.g., Rock Concert, Comedy Show).
- Extract the accessibility of the event venue (e.g., how many minutes it takes from the nearest station).
- Extract hour and date information about event, if no time is mentioned, assume full days.
- Strictly follow the following output format.

Output Format:
[Event title] {String. Title of the event.}
[Overall description] {String. Summarized description of the event, which MUST be in Japanese}
[Event category] {String. List up the event categories that are specified in the original information or identified by you.}
[Schedules] {String. List up the event date and hours that are specified in the original information.}
[Ticket price] {Integer. If the original information includes the ticket prices, please fill in this item. If not, please fill this with N/A.}
[Venue name] {String. Venue of the event.}
[Access to the venue] {String. Describe the accessibility to the event venue. Output only the facts. If not mentioned by the original information, write unknown.}
```

(a) Prompt to extract event contents.

```
We will now show you information about the events that take place at the event venues listed above. Based on this, please prepare a description of the event venue.

The following points MUST be observed.
- Create a description based only on facts. Never include lies or imaginary information in your description.
- Describe the location where the event is usually held.
- Include how to get there and how long it takes if you know, e.g., a 10-minute walk from the nearest station.
- If possible, include information that can be used as an indicator of crowding, such as the capacity size of the venue.
- Description MUST be in Japanese.
- The format of the output should be as follows

[Venue Summary] {Description of the event venue that you researched and created}
[Usual Events] {Description of the common events at the event venue that you researched}
[Venue Access] {Description of how to get to the venue you researched. If unknown, write "Unknown" }
[Crowding Indicator] {Information that can be used as an indicator of crowding at the venue you surveyed, such as the size of the venue's capacity. If unknown, write "Unknown" }
```

(b) Prompt to extract the venue information.

Figure 3: Prompting text that we utilized to extract the event contents and venue information in our experiment.

3.1.2 Feature Setting. We used four input features: event information $\boldsymbol{d}_1$, venue information $\boldsymbol{d}_2$, contextual feature $\boldsymbol{d}_3$, and time feature $\boldsymbol{d}_4$ (i.e., $N = 4$). As described in Section 2, we generate the event-level feature embedding $\boldsymbol{e}_1$ from $\boldsymbol{d}_1$ and the venue-level feature embedding $\boldsymbol{e}_2$ from $\boldsymbol{d}_2$. To create $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$, we first extracted the information mentioned in Section 2.2.1 and Section 2.2.2 through the Chat Completion API[3] with OpenAI's GPT-4 [1] (namely `gpt-4-turbo`). We used the prompts as shown in Fig. 3(a)

[2] https://infomotion.co.jp/
[3] https://platform.openai.com/docs/guides/text-generation/chat-completions-api

and Fig. 3(b) for $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$, respectively. The outputs of the prompting were converted into 1,536-dimensional embedding vectors by using the Embedding API[4] with the `text-embedding-ada-002` model, also released by OpenAI. Thus, $\boldsymbol{e}_1, \boldsymbol{e}_2 \in \mathbb{R}^{1536}$. To create the contextual feature $\boldsymbol{d}_3$, we used three pieces of information: day-of-the-week, is-holiday-or-not, and weather, which were concatenated into a single vector. Day-of-the-week was a 7-dimensional vector with one-hot encoding. Is-holiday-or-not was a 2-dimensional vector. Weather data was collected from the Japan Meteorological Agency's website. The weather was divided into four categories: {sunny, cloudy, rainy, or severe}. Thus, the weather factor was a 4-dimensional vector with one-hot encoding. Thus, $\boldsymbol{d}_3 \in \mathbb{R}^{13}$. The time length of τ is set to an hour. Thus, we formulate the time features $\boldsymbol{d}_4$ as a 24-dimensional one-hot vector.

3.1.3 Model Structure and Training Setting. To generate the embedding vectors $\boldsymbol{e}_3$ and $\boldsymbol{e}_4$ for the contextual feature $\boldsymbol{d}_3$ and time feature $\boldsymbol{d}_4$, we used a three-layer MLP with 128-dimensional outputs in the hidden space, followed by ReLU activation. We also used an MLP with the same architecture and activation to regress the mean parameter $\lambda_{l,\tau}$ from the concatenated embedding feature $\boldsymbol{e}$. For model training, we adopted a batch size of 256 and used the Adam optimizer [11] with a learning rate of 2×10^{-4}. We conducted the experiment with the leave-one-event-out (LOEO) cross-validation scheme; that is, one event was treated as the test data, and the rest of the data was treated as the training data, and this process was iteratively conducted for all the events. Early stopping was used in the model training for each iteration in the LOEO.

3.1.4 Performance Measurement. We used the mean absolute error (MAE) calculated for all time steps in the test dataset, and the correlation coefficient (CORR) calculated between the peak event crowd and its prediction for each event day.

3.1.5 Compared Methods. We compared the forecasting performance of EventOutlook with those of the combinations of three predictive models and two settings of input features in order to evaluate the performance of each method fairly. We compared the EventOutlook framework with the following baselines, including the state-of-the-art event crowd forecasting methods: (1) **Historical Average (HA)**, **Symbolic Regression (SR)** [12], and **Linear Regression (LR)** [2]. Although the existing ECF methods used manually designed features for sports events, such features cannot be applied to non-sport events. Moreover, there are no open datasets that include these features. Therefore, we compared the following feature settings: **CUSTOM**, which followed [12] in a form that is applicable to non-sport events, and **LCIF**, the proposed features.

3.2 Experimental Results

Table 1 compares the overall performance of EventOutlook with those of the baseline methods. In summary, EventOutlook improved the performance by 22.9% in MAE and 35.0% in CORR, compared to the best-performing baseline. The results show that SR performed poorly compared to the other baselines, especially in the case of SR+LLM. This unstable prediction might stem from the fact that SR increases the search space exponentially with respect to the total dimension of input features, which makes it find the optimal model structure. The dimension of the LCIF feature was about 100 times larger than the CUSTOM feature, which might result in the poor performance of SR+LLM. EventOutlook outperformed all the baselines, even if the baselines adopted the LLM feature. This result indicates the significance of leveraging event announcements and adopting the simple MLP-based model.

Table 1: Overall performance comparison.

Method			MAE↓	CORR↑
Name	**Feature**	**Model**		
HA	-	-	115.5	0.693
SR+CUSTOM	CUSTOM	SR [12]	193.0	0.493
SR+LLM	LCIF	SR [12]	197.5	0.691
LR+CUSTOM	CUSTOM	LR [2]	142.8	0.343
LR+LLM	LCIF	LR [2]	138.8	0.529
EventOutlook	LCIF	MLP	**89.1**	**0.936**

4 CONCLUSION

In this paper, we proposed the EventOutlook framework for the task of ECF for various types of events. It uses event announcements as input features for ECF by leveraging LLMs. Experimental results using real data for various types of events validated the efficacy of EventOutlook as a successful ECF method. Compared with state-of-the-art ECF methods, EventOutlook showed a 22.9% performance improvement in MAE. As such, this study contributes to understanding the effectiveness of event announcements and LLMs in forecasting crowded events. Future work will attempt to extend the forecasting capabilities of EventOutlook to include newly seen event venues. To address this issue, we plan to investigate other geographical resources, such as POIs.

Acknowledgement

This work was partially supported by JSPS KAKENHI Grant Number 22J22725 and JST PRESTO.

REFERENCES

[1] J. Achiam et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
[2] A. Al-Buenain et al. Predicting fan attendance at mega sports events—a machine learning approach: A case study of the fifa world cup qatar 2022. *Mathematics*, 12(6):926, 2024.
[3] S. Anno et al. CityOutlook+: Early crowd dynamics forecast through unbiased regression with importance-based synthetic oversampling. *IEEE Pervasive Computing*, 22(4):26–34, 2023.
[4] S. Anno et al. Forecasting lifespan of crowded events with acoustic synthesis-inspired segmental long short-term memory. *IEEE Access*, 2024.
[5] A. G. Barilla et al. The effect of promotions on attendance at major league baseball games. *Journal of Applied Business Research*, 24(3):1–14, 2008.
[6] A. B. Bortoluzzo et al. Ticket consumption forecast for brazilian championship games. *Revista de Administração (São Paulo)*, 52:70–80, 2017.
[7] Y. Chang et al. A survey on evaluation of large language models. *ACM Trans. on Intelligent Systems and Technology*, 15(3):1–45, 2024.
[8] Coxe et al. The analysis of count data: A gentle introduction to poisson regression and its alternatives. *Journal of personality assessment*, 91(2):121–136, 2009.
[9] J. Feng et al. Learning to simulate human mobility. In *Proc. of SIGKDD*, 2020.
[10] D. Getz and S. J. Page. *Event studies: Theory, research and policy for planned events.* Routledge, 2019.
[11] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
[12] G. H. Yamashita et al. Customized prediction of attendance to soccer matches based on symbolic regression and genetic programming. *Expert Systems with Applications*, 187:115912, 2022.

[4]https://platform.openai.com/docs/guides/embeddings