Understanding Event Predictions via Contextualized Multilevel Feature Learning

Songgaojun Deng
Stevens Institute of Technology
Hoboken, New Jersey, USA
sdeng4@stevens.edu

Huzefa Rangwala
George Mason University
Fairfax, Virginia, USA
rangwala@cs.gmu.edu

Yue Ning
Stevens Institute of Technology
Hoboken, New Jersey, USA
yue.ning@stevens.edu

ABSTRACT

Deep learning models have been studied to forecast human events using vast volumes of data, yet they still cannot be trusted in certain applications such as healthcare and disaster assistance due to the lack of interpretability. Providing explanations for event predictions not only helps practitioners understand the underlying mechanism of prediction behavior but also enhances the robustness of event analysis. Improving the transparency of event prediction models is challenging given the following factors: (i) multilevel features exist in event data which creates a challenge to cross-utilize different levels of data; (ii) features across different levels and time steps are heterogeneous and dependent; and (iii) static model-level interpretations cannot be easily adapted to event forecasting given the dynamic and temporal characteristics of the data. Recent interpretation methods have proven their capabilities in tasks that deal with graph-structured or relational data. In this paper, we present a Contextualized Multilevel Feature learning framework, **CMF**, for interpretable temporal event prediction. It consists of a predictor for forecasting events of interest and an explanation module for interpreting model predictions. We design a new context-based feature fusion method to integrate multiple levels of heterogeneous features. We also introduce a temporal explanation module to determine sequences of text and subgraphs that have crucial roles in a prediction. We conduct extensive experiments on several real-world datasets of political and epidemic events. We demonstrate that the proposed method is competitive compared with the state-of-the-art models while possessing favorable interpretation capabilities.

CCS CONCEPTS

• **Information systems** → **Data mining**; • **Computing methodologies** → *Knowledge representation and reasoning*; *Temporal reasoning*.

KEYWORDS

Event Prediction, Multilevel Feature Learning, Temporal Explanation

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482309

ACM Reference Format:
Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2021. Understanding Event Predictions via Contextualized Multilevel Feature Learning. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482309

1 INTRODUCTION

Human events such as flu outbreaks, protests, and crimes often have significant impacts and exhibit particular patterns. Understanding societal events and their recurring patterns is important for many stakeholders in resource allocation, personnel logistics and scheduling, and government response. With the availability of multiple media sources, data-driven machine learning methods have been widely studied in applications such as epidemic forecasting [10] and civil unrest predictions [28]. Temporal event data such as Integrated Crisis Early Warning System (ICEWS) [4] can be organized in multiple levels. At a granular level, we collect documents such as news articles which contain semantic information of events. Each event is extracted from an authentic news article and organized in structured data with relational features such as subject, action, and object as shown in Figure 1. At a high level, aggregated information such as daily counts of different types of events in a location can be calculated from these documents. Researchers have proposed to use frequency information such as event occurrence rate [14], or the number of tweets [46, 47], to predict societal events. However, frequency-based features are limited in providing easy-to-understand explanations because they are often unitary and independent. Thus, recent research has introduced sophisticated features such as graphs and text summaries to provide explainable indicators. These methods focus on identifying supporting evidence corresponding to target events of interest, such as precursor documents [25, 26] and relational graphs [8]. Recently, a systematic framework Glean [9] has been proposed to forecast event participants as well as future events. The evidence identified in existing work is either unilateral or requires additional manual screening. For example, only providing relevant news articles [25] or actor names [9] is less informative because one news article may involve multiple events simultaneously. Actor information might be limited to general terms (e.g., protester, activist). Combining various features will enable us to provide comprehensive explanations and capture hidden structures in context. However, achieving this goal brings some challenges:

- **Cross-utilizing multilevel heterogeneous data**. Deep models involving heterogeneous data (i.e., data with a variety of types and formats) have shown impressive performance in prediction

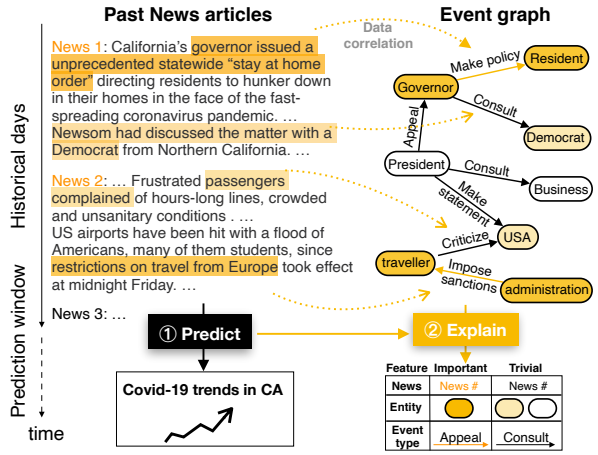


Figure 1: A motivating example of predicting and explaining the Covid-19 trend in California. Subgraphs with dark yellow node and edges are identified as key events. Highlighted text refers to identified event contexts.

tasks [9, 45]. Various formats of data are usually complementary and can provide diversified and comprehensive interpretation. However, modeling the dependencies in heterogeneous data is a new challenge in interpretable event prediction given latent correlations among features as shown in Figure 1.

- **Dependencies in temporal data**. Temporal information is essential for predicting future events [8, 13, 25]. Providing explanations for temporal data poses new challenges. Identifying a sequence of explanatory features in chronological order requires distinguishing important features at different time steps.
- **Providing multilevel explanations**. Existing interpretable approaches focus on generating or identifying static input patterns for specific tasks (e.g., graph classification) or examples (e.g., images). To obtain explanations for dynamic and temporal data, it is necessary to model the underlying associations between elements at different levels and time steps.

We address the above challenges by proposing an explainable event forecasting framework that consists of an event predictor and an event explainer. The motivation example shown in Figure 1 illustrates the prediction and interpretation process of the proposed framework. We utilize news articles and events as main input data to predict future events of interest (e.g., Covid-19 trends or protests). Events and articles are inherently related because events are extracted from the unstructured text of articles. Therefore, we model dependencies in heterogeneous data in a multilevel manner. In this example, the model predicted that newly reported Covid-19 positive cases would increase in California at a future time, and some influential elements were captured, such as "passengers complain about lines" and "travellers flooded in US airports". All the evidence suggested that this epidemic was getting worse. Acknowledging the delayed effect of policies and the long incubation period of the disease, it is understandable for the model's prediction of increasing cases in the near future. Such explanation provides information from different aspects and complements each other, which satisfies the comprehensive expectation of model explanation. Our contributions are summarized as follows:

- We propose an interpretable event forecasting framework that predicts events in the future and provides example-level explanations that can streamline the process of event analysis. The proposed framework includes (1) a predictor that models multilevel contextualized features in a hierarchical structure and processes temporal data through temporal feature learning, and (2) a post-hoc interpreter to identify main features that are related to target events at different levels.
- We propose a multilevel explanation method for event forecasting by inspecting significant elements at two levels of data. Specifically, we model the importance of semantic information in the form of documents and the effect of specific events identified from documents in top-down order.
- To deal with temporal dependency in interpretation, we introduce the concept of reference embedding that guides the generation of explanations across different time steps. The reference embedding encodes temporal attributes that are related to a model prediction. It is designed to help the model identify important features in temporal data.

The proposed method is evaluated on real-world political and epidemic event datasets compared with several state-of-the-art models. We demonstrate the ability of the proposed method in both event prediction and explanation with extensive experiments.

2 RELATED WORK

2.1 Interpretable Machine Learning

Interpretable machine learning has been widely explored given that many machine learning models are inherently a black box [11, 12, 21]. Traditional interpretable models can be divided into two main categories: (1) in-processing interpretable methods and (2) post-hoc interpretable methods. In-processing interpretable methods generate explanations in the process of decision making or while being trained, e.g., Decision Trees [30] and Attention Networks [1, 38, 39]. Post-hoc interpretable methods aim to explain the decision-making process of black box models after they are trained. Gradient-based methods such as Saliency Map [34] have been introduced to inspect feature importance via backpropagating gradient-like signals [32, 35]. Another post-explanatory study is to approximate the decision boundary of a model through feature perturbations such as LIME [29] or feature masking like L2X [5]. LIME [29] generates local sparse explanations of black box models via local surrogate interpretable models. L2X learns neural networks as an interpreter to generate feature masks, with the goal of maximizing the mutual information of masked features and predictions. In addition to grid-like data, interpretable graph neural networks have been studied to provide explanatory subgraphs for instances [22, 43] or classes [44]. In knowledge graph research, reasoning methods have been proposed to use sampling and attention mechanisms to predict future links on large-scale temporal knowledge graphs [16, 42]. However, most of these methods focus on static analysis of input features and ignore temporal dependencies in sequential data.

2.2 Interpretable Event Forecasting

Event forecasting has been broadly studied in many real-world applications such as election predictions [36], stock market movements [2], disease outbreaks [10, 33] and criminal activities [41]. Traditional existing machine learning methods work on euclidean or grid-like data. For example, frequency (and quantity) in social media streams are utilized in linear regression models to predict the occurrence time of future events [2]. More advanced features such as topic-related keywords [41], paragraph embeddings [25, 26], and semantic graphs [8, 9] have also been investigated in a variety of techniques including multi-task learning [26, 47], multi-instance learning [25], and graph representation learning [8, 9].

Identifying precursors for events is used for interpretive narrative generation and storytelling algorithms [17]. A multi-instance learning approach was proposed to jointly tackle the problems of identifying evidence-based document precursors and forecasting events of interest [25]. Key sentence identification was investigated in a multi-instance convolutional neural network (CNN) for event detection [40]. Modeling document precursors by considering the spatiotemporal effect of event-related indicators is further proposed in event forecasting [26]. Recently, graph-based event precursors have been studied in event forecasting. Deng et al. [8] introduced a dynamic graph convolutional network to encode temporal text features into graphs for forecasting societal events and identifying their context graphs. The evidence for event prediction generated by current methods is of a single type and expects additional supportive information. Therefore, these methods have limitations in identifying key information in multi-level data, which can help interpret predictions from multiple aspects.

3 PROBLEM FORMULATION

The objective of this study is to forecast the occurrence of certain categories of events (e.g., protest, flu outbreaks) and provide example-level explanations.

First, we build an event predictor $F : \mathcal{X} \rightarrow \mathcal{Y}$ that takes a sequence of historical data $X^{\leq t} = \{X^1, ...X^t\} \in \mathcal{X}$ in a location and estimates the occurrence probabilities of a set of categories of events in the future $Y^{t+\Delta} \in \{0, 1\}^M \in \mathcal{Y}$ for this location. $\Delta \geq 1$ is the lead time which denotes the number of time steps in advance for a prediction. $Y^{t+\Delta}[i] = 1$ denotes the occurrence of i-th event category. When we predict only one type of events, it becomes a binary classification problem. $X^{\leq t}$ denotes the historical data at a location before time t. Specifically, each input instance includes three types of features: (1) count vector $\{\mathbf{c} \in \mathbb{Z}^{N_e}\}^{\leq t}$ represents the occurrence frequency of N_e types of encoded events at time step t. Common event types in the domain of societal events include *protest*, *fight*, *appeal* etc.; (2) documents $\{D\}^{\leq t}$ (e.g., news articles) contains semantic information of these events, and (3) event graph $\{G\}^{\leq t}$ consists of tuples of event elements (i.e., subject, event type, object) which are extracted from documents. The input data can be simply written as $X^{\leq t} = \{\mathbf{c}, D, G\}^{\leq t}$. The encoded events in the count vector refer to the event types in a relational event graph G. We focus on predicting main categories of events (e.g., *protest*) rather than encoded event types (e.g., *engage in violent protest for policy change*).

Table 1: Important notations and descriptions.

Notations	Descriptions
M	number of labels to predict
N_e	number of event types in count vectors
w	historical time window size for event prediction
Δ	horizon/lead time of the prediction
$\mathbf{c}^t$	count vector of all encoded events at time t
$\mathbf{h}_c$	frequency embedding at time t
$\mathbf{h}_{D_i}, \mathbf{h}'_{D_i}$	document embedding (w/o and with attention)
$\mathbf{o}_{D_i}$	context embedding that involves frequency and semantic information for events extracted from i-th article
$\mathbf{a}_v, \mathbf{a}_r, \mathbf{a}_u$	embedding of subject, event type and object
$\mathbf{z}^t$	reference embedding at time t
$\mathbf{m}_D, \mathbf{m}_{(v,r,u)}$	learned news article masks and event masks

Second, we aim to explain the predictor $F : \mathcal{X} \rightarrow \mathcal{Y}$ by introducing a post-hoc event forecasting explainer $\psi : (\mathcal{X}, F) \rightarrow \mathcal{X}_s \subseteq \mathcal{X}$ to generate sequences of explanatory data. The explainer takes the input data $X^{\leq t}$ as well as latent embeddings obtained from the predictor F as input, and outputs a subset of input data as the explanation $X_s^{\leq t}$. In this work, the predictor and explainer are modules in a unified framework and are trained in order. The predictor is trained on an event forecasting task. Then, we use the predictor in evaluation mode, and use the input features and latent embeddings learned from the predictor to train the parameters in the explainer. Important mathematical notations are in described Table 1.

4 METHODOLOGY

Figure 2 provides an overview of the proposed explainable event forecasting framework which consists of (1) an event predictor module and (2) an event explainer module. The predictor has two parts: (1a) multilevel feature learning that hierarchically models heterogeneous data. It captures the dependencies between different types of data by encouraging signals to propagate from higher-level features to lower-level features. (1b) Temporal feature learning that learns sequential information across different historical time steps. The explainer is designed to provide temporal and multi-level explanations for the predictor model in evaluation mode. There are two main components in the explainer: (2a) semantic explanations which identify key documents at each historical time step. (2b) relational explanations which detect influential events (including entities and their relations) by applying edge masks.

4.1 Event Forecasting

We assume societal human events are either affected by or consequences of events in the past. In this work, instead of considering all historical information $X^{\leq t}$, we define a historical time window w and assume that the occurrences of events at time t depend on the historical events that have happened in a past window $X^{t-w+1:t}$. Specifically, we model the probability $P\left(Y^{t+\Delta}\right)$ of event occurrences at time t using the sequence $\{\mathbf{c}, D, G\}^{t-w+1:t}$ with length w:

$$P\left(Y^{t+\Delta} \middle| \{\mathbf{c}, D, G\}^{t-w+1:t}\right) = \sigma\left(\text{MLP}_\Psi\left(\mathbf{h}^t\right)\right) \in \mathbb{R}^M, \quad (1)$$

where $\mathbf{h}^t$ is the final feature vector encoding historical information from $t-w+1$ to t. Given $\mathbf{h}^t$, we utilize a multilayer perceptron with

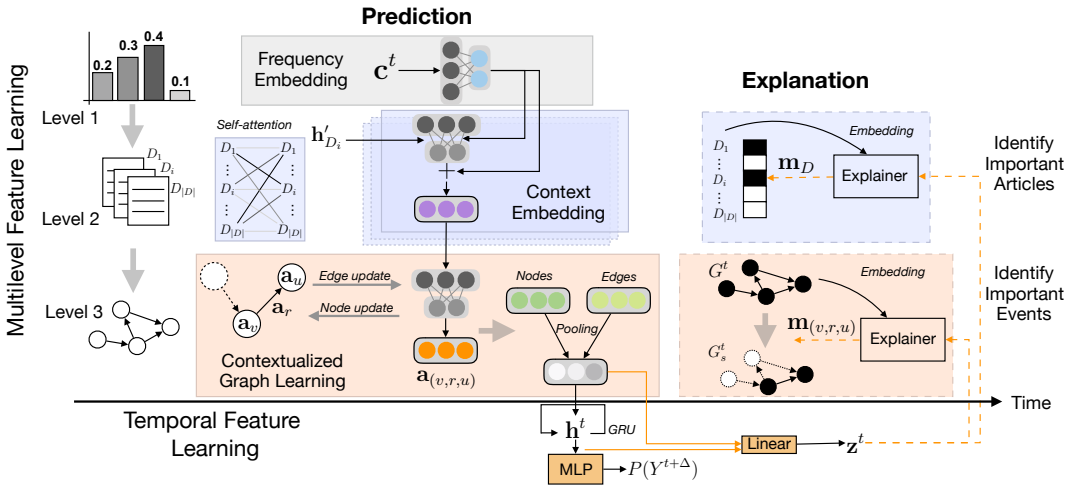


Figure 2: An overview of the proposed framework which consists of an event predictor and an event explainer. The predictor fuses hierarchical and heterogeneous data, and learns sequential information across historical time steps. The explainer provides temporal and multilevel explanations for the predictor.

one hidden layer parameterized by Ψ to obtain event probabilities where σ is an activation function (e.g., sigmoid).

To obtain the final embedding vector $\mathbf{h}^t$, we design two modules: multilevel feature learning and temporal feature learning, to fully integrate multilevel features from heterogeneous historical data.

Multilevel Feature Learning The input data consist of three types of features: event frequency, documents, and event graphs. Historical event frequencies contain high-level knowledge. For example, it shows if one type of event occurs more frequently than others. Event related documents describe detailed information such as background, participants, and summaries of events. Event graphs extracted from news articles reveal key relations among event participants. For instance, one political event can be represented as (Farmers, *protest*, Governor) in a graph. Each document contains semantic knowledge of events while event graphs only preserve event participants and actions among these participants. Multiple events can be obtained from one article. Based on the hierarchical structure of multi-level data, we introduced a top-down information modeling method for event prediction and interpretation. Specifically, we divide the multilevel feature learning process into three parts: (1) Level 1 Frequency Embedding which models event frequency information; (2) Level 2 Context Embedding which integrates frequency embeddings into text information and then serves as contextual information for specific events; (3) Level 3 Contextualized Graph Learning which propagates context embeddings into each event in an event graph.

Level 1 Frequency Embedding For each location at time t, we initially have the numerical feature $\mathbf{c}^t$ which is the count vector of all encoded events. The dimension of the count vector is fixed, and each entry represents the occurrence count of a certain type of events. Given the sparsity and the high dimensionality of this feature, we apply a linear layer $f_c(\cdot)$ to map the event count vector to a low-dimensional space $\mathbf{h}_c^t = f_c(\mathbf{c}^t)$. The frequency embedding $\mathbf{h}_c^t \in \mathbb{R}^{d_c}$ will be further integrated in the two lower level features.

Level 2 Context Embedding At this level, we first model the semantic associations between news articles and then inject the frequency embedding of events into each document. For a given location and a time step t (e.g., day), we formulate event-related documents published at time t as $D^t = \{D_1, ..., D_{|D|}\}$ and process each article into sentences, i.e., $D_i = \{s_1, ...s_{|D_i|}\}$. To obtain numerical semantic information from text, we map sentences into vector space by employing a pretrained sentence embedding model [27]. The semantic information of the i-th article can be represented as $D_i = [\mathbf{s}_1, ...\mathbf{s}_{|D_i|}]$. Sentence embedding is adopted because it encodes the meaning of a sentence, rather than discrete words. To get global semantics of an entire article, we obtain document embeddings by applying element-wise averaging on the sentence embeddings $\mathbf{h}_{D_i} = \frac{1}{|D_i|}\sum_{k=1}^{|D_i|}\mathbf{s}_k$, where $\mathbf{h}_{D_i} \in \mathbb{R}^{d_{emb}}$ is the semantic embedding for the i-th article. Articles can show potential relationships and reflect eventful clues. For example, articles may share similar topics. If a topic is mentioned in many articles, we think it is important in current context. Based on this intuition, we incorporate a self-attention mechanism [38] to capture the latent information between articles and further enhance the semantic representation of each article. Formally, we use matrix $\mathbf{H}_D$ to denote the semantic embedding of all articles collectively. Three weight matrices are applied to $\mathbf{H}_D$ to obtain Query, Key, Value matrices, respectively. We map query semantics to a set of key semantics and then output the best weighted semantic values:

$$\mathbf{H}'_D = \text{Attention}(Q, K, V) = \text{Softmax}(\frac{(\mathbf{W}_q^\top \mathbf{H}_D)^\top \mathbf{W}_k^\top \mathbf{H}_D}{\sqrt{d_k}})\mathbf{W}_v^\top \mathbf{H}_D, \quad (2)$$

where $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v \in \mathbb{R}^{d_{emb} \times d_k}$ are learnable matrices and $\mathbf{H}_D \in \mathbb{R}^{d_{emb} \times |D|}$ represents the semantic embeddings of all articles. $\mathbf{h}'_{D_i} \in \mathbb{R}^{d_k}$ is the semantic embedding of the i-th article after the attention calculation. Note that for simplicity, we have omitted t when representing document information.

Next, we concatenate the frequency embedding vector and semantic embedding vector of each article followed by a linear layer

$f_s(\cdot)$. We also include a skip connection of the frequency embedding to allow gradients to flow through multiple levels directly in the network:

$$\mathbf{o}_{D_i} = \text{ReLU}\Big(f_s(\mathbf{h}_c^t \oplus \mathbf{h}'_{D_i})\Big) + \mathbf{h}_c^t. \tag{3}$$

Here $\mathbf{o}_{D_i} \in \mathbb{R}^{d_c}$ is the context embedding for events extracted from the i-th article and $\oplus$ is concatenation. Based on the above calculation, event frequency information at time t flows to the feature of each article available at time t.

Level 3 Contextualized Graph Learning Each event extracted from a news article consists of a subject (v), an event type (r), and an object (u): $v \xrightarrow{r} u$. Events that appear in different articles may be connected when sharing the same subjects or objects. Given a location, the set of events extracted from all articles on the same day constitutes an event graph. At this level, we model relational event embeddings in the form of nodes (entities) and edges (event types) in an event graph. Next, we introduce edge embedding learning and node embedding learning.

In our problem setting, each event is provided with text knowledge besides the relational information in the event graph because events are extracted from documents (i.e, news articles). Prior work introduces a fusion method to enhance the information of entities and event types using word graphs [9]. This approach may involve random noise because words usually cannot encode important context information when only two or three hop neighbors are considered. Meanwhile, words include only local rather than global information in an article. To solve this problem, we introduce a contextualized graph learning method, which propagates event graphs by fusing higher-level global context embeddings.

For each edge in an event graph, we define a general event vector by concatenating features of subject, event type, and object. For all the events corresponding to a news article D_i, we incorporate the context embedding into each event vector. The formal steps are defined as:

$$\mathbf{a}_r = \mathbf{a}_{(v,r,u)} = \text{ReLU}\Big(f_u\big(\underbrace{\mathbf{o}_{D_i}}_{\text{context}} \oplus \underbrace{f_e(\mathbf{a}_v \oplus \mathbf{a}_r \oplus \mathbf{a}_u)}_{\text{event tuple info.}}\big)\Big), \tag{4}$$

where $f_e(\cdot), f_u(\cdot)$ are linear layers. $\mathbf{a}_v, \mathbf{a}_u \in \mathbb{R}^{d_g}$ denote subject and object embeddings. The contextual event embedding $\mathbf{a}_{(u,r,v)}$ is not fixed for the event type r, as it encodes information from the event tuple and the context information ($\mathbf{o}_{D_i}$) from the document. Here, we use contextual event embeddings to update the edge embedding, which also means that the contextual event information is stored in the edge. Then, we update node features by using the contextual event embedding in a graph message-passing network. Specifically, we perform composition ϕ on a neighboring node v with respect to the edge r using a similar method in CompGCN [37] but with contextual event embedding:

$$\mathbf{a}_u^{(l+1)} = \text{ReLU}\Big(\sum_{(v,r)\,\exists (v,r,u)\in G^t} \mathbf{W}^{(l)}\phi\big(\mathbf{a}_v^{(l)}, \mathbf{a}_{(v,r,u)}\big)\Big), \tag{5}$$

where $\mathbf{W}^{(l)} \in \mathbb{R}^{d_g \times d_g}$ is the weight matrix at the l-th propagation layer. ϕ is a non-parametric composition operator (e.g., multiplication). At the first layer $\mathbf{a}_v^{(0)} = \mathbf{a}_v$. Note that edge embedding vectors are updated only once, while node embedding vectors are updated by including information from multi-hop neighbors. At this level, the contextual information from higher-level data is integrated into relational embedding learning, so that the edge and node embeddings in an event graph are more informative.

To reduce the spatial size of representations and obtain salient features, we apply a max pooling layer over the embedding of the nodes and edges respectively:

$$\bar{\mathbf{u}} = \underset{k=1}{\overset{N_u}{\text{Max}}}\big(\mathbf{a}_{u=k}\big), \quad \bar{\mathbf{r}} = \underset{k=1}{\overset{N_r}{\text{Max}}}\big(\mathbf{a}_{r=k}\big), \tag{6}$$

where N_u, N_r denote the size of nodes and edges in an event graph.

Temporal Feature Learning After we learn contextualized relational embeddings of nodes and edges (denoted as $\{\bar{\mathbf{u}}, \bar{\mathbf{r}}\}^{t-w+1:t}$) from time $t-w+1$ to time t, we model the temporal dependency of features. We employ a Gated Recurrent Units (GRU) model [6] to obtain the final embedding vector for forecasting:

$$\mathbf{h}^t = \text{GRU}\Big(\bar{\mathbf{u}}^t \oplus \bar{\mathbf{r}}^t, \mathbf{h}^{t-1}\Big), \tag{7}$$

where $\bar{\mathbf{u}}^t \oplus \bar{\mathbf{r}}^t$ is the input at time t, and $\mathbf{h}^{t-1} \in \mathbb{R}^{d_f}$ is the hidden state at time $t-1$. The final hidden state $\mathbf{h}^t$ is then utilized in the final prediction in Eq. 1. Other recurrent networks can be applied.

Optimization We compare predictions with ground truth observations and optimize the cross entropy loss across different labels:

$$\mathcal{L} = -\sum_{n}^{N}\sum_{m}^{M} Y_n[m]\log\hat{Y}_n[m] + \Omega(\Theta), \tag{8}$$

where Y is the ground truth label and $\hat{Y} = P(Y)$ is the model prediction. The time superscript is omitted here. M, N represents the number of labels and samples. $\Omega(\Theta)$ stands for the ℓ_2-norm regularization for all training parameters. All the model parameters will be trained and optimized by back-propagation.

Understanding the behavior of a model helps practitioners evaluate predictions and the trustworthiness of the model [29]. Next, we introduce a new multilevel temporal interpretation method to explain our event predictor.

4.2 Multilevel Explanation

To explain predictions made by the event predictor F, we propose an explainer module to provide multilevel explanations on both text (i.e., news articles) and multi-relational graphs (i.e., events).

The learning objective Given an instance, our goal is to identify a subset of news articles $D_s \in D$ and a subgraph $G_s \in G$ of events over the historical time window $(t-w+1:t)$ that are important for the event predictor's prediction $\hat{y}^{t+\Delta}$. Following prior work [5, 22, 43], we formalize the notion of importance using mutual information (MI) and formulate our explanation module as the following optimization framework:

$$\begin{aligned}\underset{\{D_s,G_s\}^{t-w+1:t}}{\text{Max}}\ & MI\Big(\hat{y}^{t+\Delta}, \{D_s, G_s\}^{t-w+1:t}\Big) \\ &= H\Big(\hat{y}^{t+\Delta}\Big) - H\Big(\hat{y}^{t+\Delta}\Big|\{D_s, G_s\}^{t-w+1:t}\Big),\end{aligned} \tag{9}$$

where $\hat{y}^{t+\Delta}$ denotes the predicted class of the event model with $\{\mathbf{c}, D, G\}^{t-w+1:t}$ as the input. H denotes the entropy. $\{D_s, G_s\}^{t-w+1:t}$ is the explanatory results in the form of time series. Given an instance, the mutual information quantifies the change in the probability of prediction $\hat{y}^{t+\Delta} = F(\{\mathbf{c}, D, G\}^{t-w+1:t})$ when the input news article data are limited to $\{D_s\}^{t-w+1:t}$ and the event graphs

are limited to $\{G_s\}^{t-w+1:t}$. Direct approximation of the above objective function is intractable since it requires summing over all combinations of feature subsets. We consider a relaxation by using parameterized neural networks to learn the scores of news articles and events. This operation is similar to "attention" [1] where the weight of each feature is parametrized by a function of the respective feature itself.

To build the interpretation network, we first introduce *reference embedding*, which is regarded as a guideline for selecting important features at each historical time step. $\mathbf{h}^t$ is the global latent representation used for the final event prediction (Eq. 1). Intuitively, we assume $\mathbf{h}^t$ encodes important information about historical events. $\{\bar{\mathbf{u}}^t, \bar{\mathbf{r}}^t\}$ are input of temporal feature learning. Based on this information, we define the reference embedding at time τ as a linear transformation $f_{z_1}(\cdot)$ of the concatenation of $\mathbf{h}^t$ and $\{\bar{\mathbf{u}}^t, \bar{\mathbf{r}}^t\}$:

$$\mathbf{z}^\tau = f_{z_1}(\mathbf{h}^t \oplus \bar{\mathbf{u}}^\tau \oplus \bar{\mathbf{r}}^\tau), \quad t - w + 1 \leq \tau \leq t. \tag{10}$$

Identifying key articles Given the collection of news articles at time t, we obtain the vector representation $\mathbf{h}_{D_i}$ for an article from the predictor. To evaluate the importance of each article, we propose to calculate a score for each article based on the semantic embedding of the article and the reference embedding. Formally, we employ the additive attention function [1]:

$$\alpha_{D_i} = \mathbf{v}^\top \text{Tanh}\big(\mathbf{W}_a(\mathbf{h}_{D_i} \oplus \mathbf{z}^t)\big), \tag{11}$$

where $\mathbf{v}, \mathbf{W}_a$ are learnable parameters of the explainer. To binarize the scores for feature selection, we first use Sparsemax [23] to normalize the scores to a sparse distribution. Sparsemax was used for sparse attention weights [7]. Given the vector form of the scores $\boldsymbol{\alpha}_D$ of all articles at time t, the formal binary transformation is:

$$\mathbf{m}_D = \text{Tanh}\big(\text{Sparsemax}(\boldsymbol{\alpha}_D)/\epsilon\big), \tag{12}$$

where $\epsilon = 1e-12$ is a small value to enforce a positive value to 1. $\mathbf{m}_D$ is the learned news article mask. Thus, articles at each timestamp are reduced from D to D_s. To optimize the explainer with the selected features, the columns representing the articles that are masked in $\mathbf{H}_D$ will be filled with zero vectors.

Identifying key events In an event graph, nodes are entities and edges are event types, and each connection represents an event. To identify important events, we focus on selecting edges in a graph and connected nodes are automatically considered to be important. Specifically, we first define an event embedding as a linear transformation $f_{ex}(\cdot)$ of its edge embedding and embeddings of its two connected nodes. Then, we apply a nonlinear transformation on combined reference embeddings and event embeddings to evaluate the importance of this event:

$$\alpha_{(v,r,u)} = \text{Tanh}\Big(f_{z_2}\big(\mathbf{z}^t \oplus f_{ex}(\mathbf{a}_v \oplus \mathbf{a}_r \oplus \mathbf{a}_u)\big)\Big). \tag{13}$$

Here $f_{z_2}(\cdot)$ is a linear layer and the Tanh function regularizes the input value. To obtain event masks, we employ the same method as above to binarize the scores. To reflect multi-level attributes while evaluating an event, we also include the article score from which the event is extracted. We expect that events extracted from important articles have a higher chance to be identified. The steps are defined as follows:

$$\mathbf{m}_{(v,r,u)} = \text{Tanh}(\text{Sparsemax}(\boldsymbol{\alpha}_{(v,r,u)} + \boldsymbol{\alpha}_D)/\epsilon). \tag{14}$$

With the event mask $\mathbf{m}_{(v,r,u)}$, the event graph G is reduced to G_s. After learning the masks for multiple time steps, we obtain the sequence of explanatory data $\{D_s, G_s\}^{t-w+1:t}$. These data are fed to the event predictor together with event frequency data $\{\mathbf{c}\}^{t-w+1:t}$ for optimizing the objective function (Eq. 9). The trainable parameters are $\mathbf{v}, \mathbf{W}_a$ and the weights in layers of $f_{z_1}(\cdot), f_{z_2}(\cdot), f_{ex}(\cdot)$.

Table 2: Dataset Statistics. Loc. indicates the number of locations the events are selected. GEO represents the geographic granularity of the data. Note that each sample in our datasets refers to one location.

Dataset	Loc.	Events	News	Entities	GEO	Samples
India	5	138,202	84,616	3,246	City	5,468
Russia	2	134,635	70,369	4,654	City	2,466
Thailand	5	99,932	52,424	2,491	City	2,505
Egypt	5	268,364	71,049	2,977	City	2,029
Covid-19	10	410,240	307,346	3,465	State	2,523

5 EXPERIMENTAL SETUP

5.1 Datasets

We prepare five datasets from two data sources for evaluation: four political event datasets and one epidemic event dataset. Among the four political event datasets, two of them, India (IN) and Russia (RU), contain political events from 01/01/2012 to 12/31/2016 collected from Integrated Conflict Early Warning System (ICEWS) [4]. The other two, Thailand (TH) and Egypt (EG), are collected from Global Database of Events, Language, and Tone (GDELT) [20] ranging from 01/01/2017 to 12/31/2019. These events are encoded into 20 main categories (e.g., protest, fight, consult) using Conflict and Mediation Event Observations (CAMEO) [3] event codes. Each event has attributes such as geolocation, time (day, month, year), category, entity (subject, object) and its associated text. Event text is available in the ICEWS data source. For GDELT, we manually crawl news articles using provided event source URLs.[1] In the experiments, we focus on predicting one type of events with significant impact: *Protest*. Thus this task becomes a binary classification problem. The epidemic event dataset, Covid-19 (C-19), is used for predicting the trend of newly reported Covid-19 cases in the United States.[2] The ground truth data include the average daily new case growth of every 7 days from 03/15/2020 to 12/08/2020 in 10 states. We categorize numerical values into three classes and perform a multi-class classification task. A growth rate greater than 8% is a *substantial increase*, less than 0% is a *decline*, and others imply *small growth*. The proportions of the three types of samples are 0.0959, 0.4598, and 0.4443 respectively. The input data are obtained from GDELT, maintaining the same format as other political event datasets. We select news articles related to Covid-19 by filtering with a set of predefined keywords, e.g., coronavirus and quarantine. The data statistics are shown in Table 2.

5.2 Evaluation Metrics

We use the following metrics to evaluate our method:

[1] https://github.com/codelucas/newspaper
[2] https://github.com/nytimes/covid-19-data

Table 3: Prediction results showing the mean and standard deviation of F1 score on all datasets. Bold denotes the best and underline denotes the second best.

Method	Thailand	Egypt	India	Russia	Covid-19
DNN	0.643 (.02)	0.631 (.01)	0.609 (.01)	0.568 (.06)	0.504 (.03)
RGCN	0.629 (.04)	0.605 (.02)	0.578 (.02)	<u>0.616</u> (.01)	0.575 (.05)
CompGCN	0.648 (.02)	0.524 (.05)	0.573 (.03)	0.589 (.03)	0.527 (.03)
GRU	0.633 (.04)	0.648 (.01)	0.605 (.01)	0.578 (.03)	0.593 (.02)
DynGCN	0.580 (.02)	0.509 (.02)	0.607 (.03)	0.597 (.02)	-
RENET	0.612 (.05)	0.648 (.02)	0.595 (.02)	0.578 (.03)	0.574 (.03)
Glean	0.665 (.02)	0.653 (.03)	<u>0.614</u> (.01)	0.596 (.01)	0.616 (.02)
CMF	<u>0.683</u> (.01)	<u>0.675</u> (.01)	**0.621** (.01)	**0.623** (.01)	<u>0.645</u> (.01)
$\textbf{CMF}_{\text{w/o freq}}$	**0.689** (.01)	0.652 (.02)	0.607 (.01)	0.606 (.01)	**0.650** (.02)
$\textbf{CMF}_{\text{w/o news}}$	0.657 (.04)	**0.681** (.01)	0.601 (.03)	0.599 (.02)	0.584 (.02)
$\textbf{CMF}_{\text{w/o event}}$	0.614 (.05)	0.652 (.02)	0.589 (.02)	0.555 (.03)	0.574 (.01)

Prediction Performance. We use F1 score to evaluate the performance of event prediction. For the Covid-19 dataset, which is formulated as a multi-class classification task, we consider the weighted averaging of F1 scores over all classes.

Explanation Performance. we adopt Accuracy and Area under the ROC Curve (AUC) to quantitatively evaluate the proposed explanation method and conduct case studies to further demonstrate the effectiveness of our proposed method in interpretation.

5.3 Comparative Methods

We compare our event forecasting method with several state-of-the-art baselines with different features: (1) Models using event frequency features: Deep Neural Networks (**DNN**), **GRU** [6]; (2) Models using graph based features (word graphs or event graphs): **RGCN** [31], **CompGCN** [37], **DynGCN** [8], **RENET** [18], **Glean** [9]. Note that **DynGCN** is a protest prediction model, so we do not compare this model on the Covid-19 dataset. For static models (**LR**, **DNN**, **RGCN**, **CompGCN**), the input is the accumulation of data in historical time steps. To estimate the importance of multi-level features in our model, we vary our base model in three ways: $\textbf{CMF}_{\text{w/o freq}}$ removes the top-level event frequency embedding, $\textbf{CMF}_{\text{w/o news}}$ removes the middle-level features, and $\textbf{CMF}_{\text{w/o event}}$ deletes the bottom level and uses the pooling of context embeddings as input to the GRU model.

For explanations, we compare our explainer with two methods that are able to interpret the proposed event predictor: A gradient (**G**) [34] method and a random (**R**) method. The gradient method directly uses gradients with respect to events and document embeddings as feature importance. Specifically, we obtain the importance score of an event by adding the mean values of the feature vectors of subject, event type, and object in this event. Similar methods are applied for news articles. The key features are selected through the Sparsemax function (i.e., Eq. 12, 14). The random method generates random masking scores for events and news articles.

5.4 Parameter Setting

To obtain vector representations of text data, we pre-train two Sent2vec models [27] using all available news articles from ICEWS and GDELT, respectively. We randomly split the data into training, validation, and test sets at a ratio of 60%-20%-20% for each dataset. To reduce the sparsity of the data, we randomly remove a small part of the negative samples from the India, Russia and Thailand datasets. The positive rates of event datasets of India, Russia, Thailand and Egypt are 0.477, 0.325, 0.194 and 0.306 respectively.

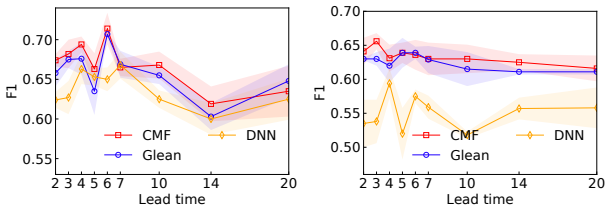
For hyperparameters, the historical time window size w is set to 7. The lead time Δ is set to 1 and 10 for political event datasets and the epidemic event dataset, respectively. The number of encoded events N_e is set to 291 according to CAMEO event codes. The dimension of sentence embedding d_{emb} is 300. The feature dimensions d_c, d_f, d_g are set to be the same, and are grid searched from $\{64, 128\}$. The feature size for semantic information d_k is searched from $\{32, 64\}$. The number of layers for node embedding learning in event graphs (Eq. 5) is searched from $\{1, 2\}$. We employ 1-layer GRU [6] as our temporal feature learning. All parameters, including the embeddings of all entities and event types, are initialized with Glorot initialization [15] and trained using the Adam [19] optimizer with weight decay 1e-5 and dropout rate 0.5. The learning rate is searched from $\{0.001, 0.003, 0.005\}$. The batch size is 32 for all datasets. We set the maximum number of epochs to 25. Our model usually converges within 10 epochs for political event datasets and 20 epochs for epidemic event datasets. The best models are selected by early stopping when the F1 score does not decrease for 5 consecutive epochs. All experimental results are the average of 5 randomized trials. All code is implemented using Python 3.7.9 and Pytorch 1.7.0 with CUDA 9.2.

6 EXPERIMENTAL RESULTS

6.1 Results of Event Prediction

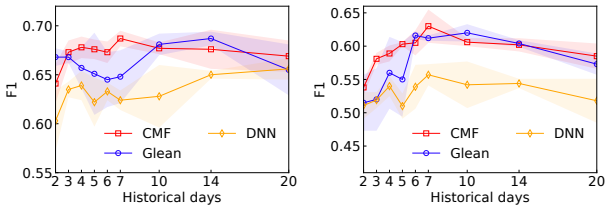
The F1 scores of event forecasting tasks on all datasets are shown in Table 3. We divide all models into static and temporal models based on their ability to model time dependencies. We observe that the predictive performance of temporal models is overall better than static models, especially on Egypt and Covid-19 datasets. Models using event frequency data (**DNN**, **GRU**) have achieved good F1 scores on Thailand, Egypt and India datasets, demonstrating the importance of past event frequency patterns. Methods using event graphs in a static or temporal manner also have good F1 values (**RGCN**, **CompGCN**, **RENET**, **Glean**). The performance of the dynamic graph model **DynGCN** is unstable, and it does not perform well on datasets from GDELT. This is because the size of news articles and events from GDELT is much larger than ICEWS. Note that the overall F1 score for political events is lower than the results reported in the **DynGCN** paper, which is mainly due to different label settings. In the **DynGCN** paper, negative samples are constructed based on the event occurrence of target dates as well as the previous three days. We relax this setting by ignoring the three-day historical constraint. The proposed method involving multilevel feature learning outperforms the baselines on all datasets in F1 score. The results suggest that the introduction of multiple types of features plays a key role in improving prediction scores.

6.1.1 Ablation Study. Results of the ablation study are shown at the bottom of Table 3. Comparing the variants of our method, we observe that removing lower level features leads to non-trivial decreases of F1, which indicates the importance of detailed information in event prediction. The variant $\textbf{CMF}_{\text{w/o freq}}$ without



(a) Egypt (b) Covid-19

Figure 3: F1 score with varying lead time.



(a) Egypt (b) Covid-19

Figure 4: F1 score with varying historical days.

frequency features sometimes outperforms the full model. A possible reason is that the frequency information is less informative than other types of features. The variant $\textbf{CMF}_{\text{w/o news}}$ neglecting news article features achieves the best F1 on the Egypt dataset. This can be explained by the low ratio of the number of news to the number of events in this dataset. Many events lack source news articles, which makes contextual information less helpful. The F1 score of $\textbf{CMF}_{\text{w/o event}}$ is the worst, which shows that graph learning on event graphs contributes more to the final result. Overall, the results show the importance of multilevel feature learning in improving event prediction ability in terms of F1 score. Meanwhile, changing the size of hidden layers in our model can control the flow of information from the upper layer to the lower layer, thereby maximizing the generalization ability of the model on different datasets.

6.1.2 Sensitivity Analysis. We investigate the sensitivity of **lead time** and **historical days** in the proposed method. We report the results on Egypt and Covid-19 datasets in event prediction tasks. Two representative baselines **Glean** and **DNN** are selected.

Figure 3 shows the F1 score in different lead time settings on the two datasets. The proposed model achieves the best F1 score when the lead time is 6 days on the Egypt dataset and approximately 3 or 5 days on the Covid-19 dataset. The models' prediction performance will decrease in a longer lead time, mainly due to the information gap between the historical data and the forecasting time. Our model outperforms the baselines in most cases. We report the F1 score with varying historical days from 2 to 20 in Figure 4. We can observe that our model can achieve higher F1 scores when using more distant historical records. However, when the length is greater than 7, the F1 score stabilizes and may decrease slightly. It implies that more historical data cannot enhance the predictive ability of the model. Our model is more stable than the other two models and has lower variance, especially for shorter historical time windows.

6.2 Results of Explanation

Table 4: Performance comparison for explanation methods.

	CMF	G	R
	AUC		
TH	0.86	0.74	0.64
EG	0.82	0.63	0.57
IN	0.82	0.77	0.56
RU	0.92	0.81	0.68
	Accuracy		
C-19	0.83	0.48	0.40

6.2.1 Quantitative Evaluation. Since there are no ground truth data for explanations, we conduct an experiment to evaluate how close an explanation is to approximate the prediction of the proposed model, thereby evaluating interpretation fidelity [24]. Table 4 shows the results of three explainers by formalizing quantitative interpretation evaluation as a classification problem [22, 43]. The ground truth is the prediction from the proposed event predictor using full input data, and the prediction score is the output of the event predictor using only selected explanatory data. This evaluates how much an explainer understands the action of a classifier. We compare our explainer with a gradient method (**G**) and a random model (**R**). All explainers aim to interpret the same event predictor. We report AUC on the binary classification task and accuracy on the multi-class classification problem. We observe that the proposed explainer approximates model predictions better than other methods.

6.2.2 Case Studies. We showcase two examples of identified key articles and events from the proposed method when predicting Covid-19 trends. Note that we use input data from the past 7 days and make predictions on the 10th day in the future. As shown in Figure 5, we list part of the important features in the form of articles and event graphs. In this example, our model predicts that the number of new covid-19 cases in California will increase significantly on Mar. 30. We summarize the main topic of each article in one sentence. According to the provided explanation, new positive cases were detected in California, and the allocation of medical resources was in progress. This shows that the epidemic is getting worse. Congestion and chaos at US airports suggest the possible spread of the virus. Social distancing policies were issued, and some people were preparing to respond to covid-19 emergencies. Such information shows that people were taking action to control the virus. Because the impact of infectious viruses is usually delayed, it is easy to understand why the model predicted "increase" (*small growth*) given the provided evidence. The relational graphs identified from the proposed model capture key entities that play a pivotal role in the development of Covid-19. To show how the model interprets a "decrease" (*decline*) trend, we provide another example for New York on Aug. 29 in Figure 5. From the identified data, we can see some virtual activities as responses to Covid-19. We think people are participating in various activities online instead of on-site. This behavior effectively helps prevent the spread of the virus.

We also present an explanation example of the protest event prediction as shown in Figure 6. The information shown on the left intends to explain why the proposed model predicts that protest events are likely to occur in New Delhi, India on May 2, 2015.

6.2.3 Stability Analysis. Stability assesses the similarity of explanations for similar instances [24]. The lack of stability can lead

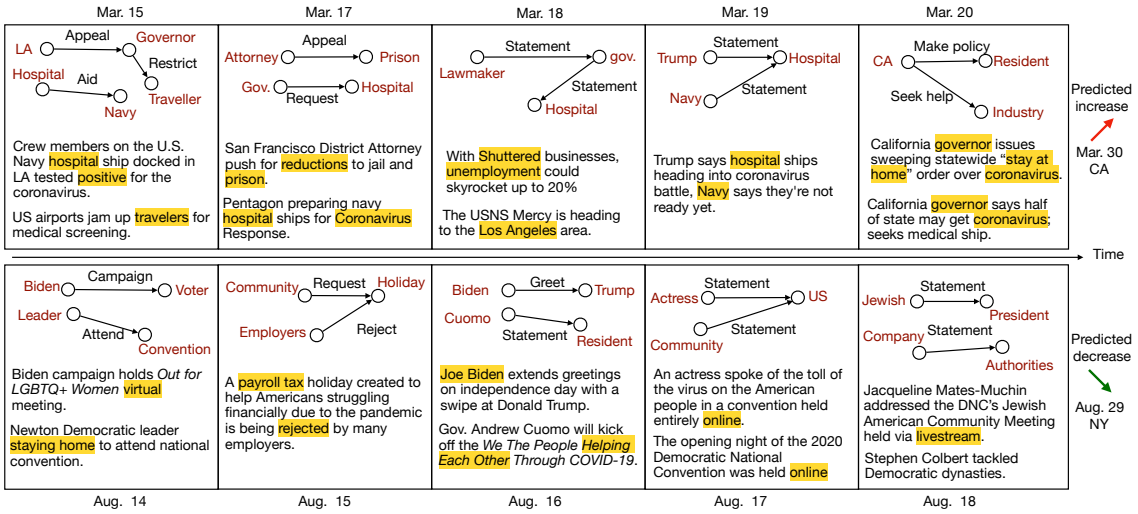


Figure 5: Two explanation examples on the Covid-19 dataset. The top one explains a prediction of *substantial increasing* trend for California on Mar. 30. The bottom one interprets a prediction of *decreased* trend for New York on Aug. 29. The highlighted phrases refer to identified event context. The selected relations capture key entities in the spread of Covid-19.

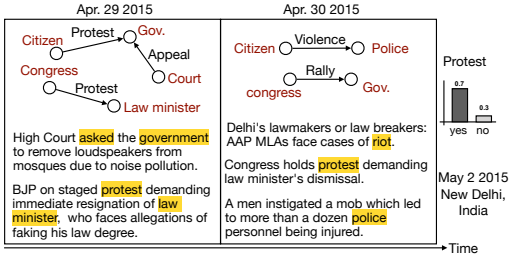
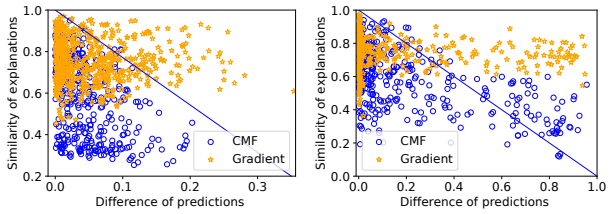


Figure 6: Case study of a protest prediction in India.



(a) Russia **(b) Thailand**

Figure 7: Stability analysis on explanation results.

to a high variance in interpretation results. In the sequential explanation of event prediction, stability property is important due to the temporal dependency of the data. For example, if the event predictions for two consecutive days are the same, in the case of overlapping historical data, the interpretations are expected to be similar. To evaluate this property of our model, we conduct an analysis on Russia and Thailand datasets. For each dataset, we collect all sample pairs that have the same location and whose predicted time differs by 1 day. Thus, for each pair of samples, they should have common data in their input window. We evaluate the similarity of the explanations of the two samples by averaging the "Intersection over Union" of news articles and events. The prediction difference between two samples is the difference in the prediction scores of the positive class. The results are shown in Figure 7, where each point represents the evaluation of a pair of similar samples. We can observe that the overall evaluation of our method follows the pattern that points with larger prediction differences have lower interpretation similarities and vice versa. The gradient method leads to a high degree of similarity of almost all points. The points in the lower-left corner are the result of sparsity in the input data given that not every historical day has data.

7 CONCLUSION AND FUTURE WORK

Interpretation is crucial in societal event prediction because it can produce supporting evidence for reliable decision-making in various fields. In this paper, we present a novel framework for forecasting future events and providing multilevel explanations for predictions. We demonstrated the effective prediction performance of the proposed model on real-world political and epidemic event datasets. We also show the interpretability of the model through quantitative analysis and real-world case studies. One of the limitations of this work is that we only model news and event information in specific locations for prediction and interpretation while ignoring the influence of the global context. Future work will consider exploring spatial dependencies in event data and explain event predictions from different locations.

ACKNOWLEDGMENTS

This work is supported in part by the US National Science Foundation under grants 1948432 and 2047843. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of computational science* 2, 1 (2011), 1–8.

[3] Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. CAMEO.CDB.09b5.pdf. In *ICEWS Coded Event Data*. Harvard Dataverse. https://doi.org/10.7910/DVN/28075/SCJPXX

[4] Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. ICEWS Coded Event Data.

[5] Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. 2018. Learning to explain: An information-theoretic perspective on model interpretation. In *ICML '18*. PMLR, 883–892.

[6] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *EMNLP* (Doha, Qatar). Association for Computational Linguistics, 1724–1734.

[7] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. 2016. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. *NeurIPS ' 16* 29 (2016), 3504–3512.

[8] Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2019. Learning Dynamic Context Graphs for Predicting Social Events. In *KDD '19*. ACM, 1007–1016.

[9] Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2020. Dynamic Knowledge Graph Based Multi-Event Forecasting. In *KDD '20*. Association for Computing Machinery, New York, NY, USA, 1585–1595.

[10] Songgaojun Deng, Shusen Wang, Huzefa Rangwala, Lijing Wang, and Yue Ning. 2020. Cola-GNN: Cross-location Attention based Graph Neural Networks for Long-term ILI Prediction. In *CIKM '20*.

[11] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).

[12] Mengnan Du, Ninghao Liu, and Xia Hu. 2019. Techniques for interpretable machine learning. *Commun. ACM* 63, 1 (2019), 68–77.

[13] Yuyang Gao, Liang Zhao, Lingfei Wu, Yanfang Ye, Hui Xiong, and Chaowei Yang. 2019. Incomplete Label Multi-Task Deep Learning for Spatio-Temporal Event Subtype Forecasting. In *AAAI*, Vol. 33. 3638–3646.

[14] Matthew S Gerber. 2014. Predicting crime using Twitter and kernel density estimation. *Decision Support Systems* 61 (2014), 115–125.

[15] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*. 249–256.

[16] Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2021. Explainable Subgraph Reasoning for Forecasting on Temporal Knowledge Graphs. In *ICLR '21*.

[17] M Shahriar Hossain, Patrick Butler, Arnold P Boedihardjo, and Naren Ramakrishnan. 2012. Storytelling in entity networks to support intelligence analysts. In *KDD '12*. 1375–1383.

[18] Woojeong Jin, Changlin Zhang, Pedro Szekely, and Xiang Ren. 2019. Recurrent event network for reasoning over temporal knowledge graphs. *arXiv preprint arXiv:1904.05530* (2019).

[19] D Kinga and J Ba Adam. 2015. A method for stochastic optimization. In *ICLR '15*, Vol. 5.

[20] Kalev Leetaru and Philip A Schrodt. 2013. Gdelt: Global data on events, location, and tone, 1979–2012. In *ISA annual convention*, Vol. 2. Citeseer, 1–49.

[21] Zachary C Lipton. 2018. The mythos of model interpretability. *Queue* 16, 3 (2018), 31–57.

[22] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. 2020. Parameterized Explainer for Graph Neural Network. *NIPS '20* 33 (2020).

[23] Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *ICML '16*. 1614–1623.

[24] Christoph Molnar. 2019. *Interpretable Machine Learning*. https://christophm.github.io/interpretable-ml-book/.

[25] Yue Ning, Sathappan Muthiah, Huzefa Rangwala, and Naren Ramakrishnan. 2016. Modeling precursors for event forecasting via nested multi-instance learning. In *KDD '16*. ACM, 1095–1104.

[26] Yue Ning, Rongrong Tao, Chandan K Reddy, Huzefa Rangwala, James C Starz, and Naren Ramakrishnan. 2018. STAPLE: Spatio-Temporal Precursor Learning for Event Forecasting. In *SIAM*. SIAM, 99–107.

[27] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In *NAACL '2018*.

[28] Naren Ramakrishnan, Patrick Butler, Sathappan Muthiah, Nathan Self, Rupinder Khandpur, Parang Saraf, Wei Wang, Jose Cadena, Anil Vullikanti, Gizem Korkmaz, et al. 2014. 'Beating the news' with EMBERS: forecasting civil unrest using open source indicators. In *KDD '14*. ACM, 1799–1808.

[29] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should I trust you?" Explaining the predictions of any classifier. In *KDD '16*. 1135–1144.

[30] S Rasoul Safavian and David Landgrebe. 1991. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics* 21, 3 (1991), 660–674.

[31] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *ESWC*. Springer, 593–607.

[32] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. *arXiv preprint arXiv:1704.02685* (2017).

[33] Alessio Signorini, Alberto Maria Segre, and Philip M Polgreen. 2011. The use of Twitter to track levels of disease activity and public concern in the US during the influenza A H1N1 pandemic. *PloS one* 6, 5 (2011), e19467.

[34] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).

[35] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. 2014. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806* (2014).

[36] Andranik Tumasjan, Timm Oliver Sprenger, Philipp G Sandner, and Isabell M Welpe. 2010. Predicting elections with twitter: What 140 characters reveal about political sentiment. *ICWSM* 10, 1 (2010), 178–185.

[37] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2019. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082* (2019).

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS '17*. 5998–6008.

[39] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[40] Wei Wang, Yue Ning, Huzefa Rangwala, and Naren Ramakrishnan. 2016. A multiple instance learning framework for identifying key sentences and detecting events. In *CIKM '16*. 509–518.

[41] Xiaofeng Wang, Matthew S Gerber, and Donald E Brown. 2012. Automatic crime prediction using events extracted from twitter posts. In *SBP-BRiMS '12*. Springer, 231–238.

[42] Xiaoran Xu, Wei Feng, Yunsheng Jiang, Xiaohui Xie, Zhiqing Sun, and Zhi-Hong Deng. 2019. Dynamically pruned message passing networks for large-scale knowledge graph reasoning. *arXiv preprint arXiv:1909.11334* (2019).

[43] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. Gnnexplainer: Generating explanations for graph neural networks. In *NIPS '19*. 9244–9255.

[44] Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. 2020. XGNN: Towards Model-Level Explanations of Graph Neural Networks. *arXiv preprint arXiv:2006.02587* (2020).

[45] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129* (2019).

[46] Liang Zhao, Feng Chen, Chang-Tien Lu, and Naren Ramakrishnan. 2016. Multi-resolution spatial event forecasting in social media. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 689–698.

[47] Liang Zhao, Qian Sun, Jieping Ye, Feng Chen, Chang-Tien Lu, and Naren Ramakrishnan. 2015. Multi-task learning for spatio-temporal event forecasting. In *KDD '15*. ACM, 1503–1512.