IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 13, 2022, accepted April 25, 2022, date of publication May 2, 2022, date of current version May 9, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3171843

A Novel Multi-Module Approach to Predict Crime Based on Multivariate Spatio-Temporal Data Using Attention and Sequential Fusion Model

NOWSHIN TASNIM, IFTEKHER TOUFIQUE IMAM, AND M. M. A. HASHEM
Department of Computer Science and Engineering, Khulna University of Engineering and Technology, Khulna 9203, Bangladesh
Corresponding author: M. M. A. Hashem (hashem@cse.kuet.ac.bd)

This work was supported in part by the Khulna University of Engineering and Technology (KUET), Bangladesh; and in part by the Department of Computer Science and Engineering, KUET Alumni.

ABSTRACT Forecasting crime is complex since several complicated aspects contribute to a crime. Predicting crime becomes more challenging because of the enormous number of everyday crime episodes in varied places. Though there are many established machine learning and deep learning techniques, law enforcement officers face challenges in preventing crime from occurring promptly. An efficient way of law enforcement is required to lower the crime rates. This paper proposed an effective multi-module method for predicting crime using deep learning techniques. Our proposed method has two modules: Feature Level Fusion and Decision Level Fusion. The first module employs temporal-based Attention LSTM, Spatio-Temporal based Stacked Bidirectional LSTM, and Fusion model. The Fusion model leverages the prior two models' training data. The temporal-based model is the source model for the transfer learning technique on the dataset of different cities. By applying this technique, the training time of the model is reduced. In the second module, the Spatio-Temporal based Attention-LSTM, Stacked Bidirectional LSTM, and the result of feature-level fusion module are used to get the final prediction. The proposed architecture predicts the next hour based on the data from the past twenty-four hours. The estimated number of crimes in any category for a particular location can be obtained as the output of our suggested model. It also enables law enforcement to get insight into future crime occurrences based on category, time, and location. This work concentrated mainly on the USA's San Francisco and Chicago cities for the experimental analysis. For the San Francisco and Chicago datasets, our model has the Mean Absolute Error of 0.008, 0.02, the Coefficient of Determination of 0.95 and 0.94, and the Symmetric Mean Absolute Percentage Error of 1.03% and 0.6%, respectively. The proposed model outperforms numerous other well-known models.

INDEX TERMS Attention LSTM, crime prediction, crime trend, fusion model, multi-module deep learning model, multivariate time series, Stacked Bidirectional LSTM.

I. INTRODUCTION

Analyzing time-series data to extract meaningful statistics and other characteristics is the main target of Time Series Analysis. This information highly dominates in predicting future values based on previously observed data. Due to continuous urbanization and growing populations, violent crimes and accidents are on the rise. Extracting information and analyzing the hidden patterns, the co-relation between these vast amounts of data through Big Data Analysis (BDA) is one of the trending approaches nowadays [1], [2]. BDA can help revolutionize how authorities maintain and protect people from crime. As the population grows, crime activity patterns become more varied and complex. It requires new approaches to understand and tackle it. Finding the pattern, predicting the future crimes, and matching these patterns with newly available data through data analysis are the main strategies to tackle the crime problem. Thus, predicting the likelihood of dealing with crime at a specific time or place will be convenient through the time-series-based BDA. The system can predict the possibility by looking for precipitating factors such as geographical location, economic condition, time of the year, and environment [3], [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Alberto Cano.

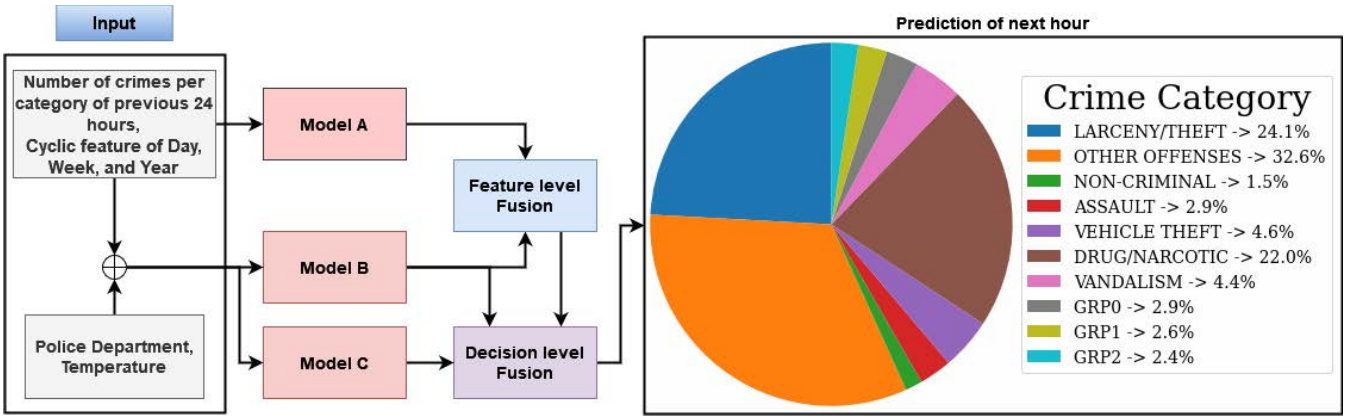


FIGURE 1. An abstract diagram of deep learning based proposed method (Model A represents ATTN-LSTM model for temporal feature, Model B represents St-Bi-LSTM model, and Model C represents ATTN-LSTM model for Spatio-temporal features).

Crime is one of the most predominant and alarming adversities in our society, and its prevention is a vital task [4]. Crime analysis and prediction is a systematic approach for analyzing and identifying different patterns, relations, and trends in crime and disorder [2]. Crime occurrences are common all over the world, mainly in cities. It hinders fundamental human rights and brings collapse to the structure of society. It is not always possible for the law enforcers to find out in which areas the crime rate is high at a specific time manually. Because the motives of occurring crime are dynamic and there is no proper utilization of existing crime data. Suppose an automated system having a higher accuracy of prediction is available to the law enforcers. In this case, they can take the necessary precautions to decrease crime to a specific rate. Some commonly used techniques to predict and forecast crime data are: logistic regression, support vector machine (SVM), Naive Bayes, k-nearest neighbors (KNN), decision tree, multilayer perceptron (MLP), random forest, and eXtreme Gradient Boosting (XGBoost), time series analysis using LSTM and autoregressive integrated moving average (ARIMA) models. In recent years, deep-learning (DL) based models are also used for forecasting purposes, such as- crime forecasting [1], [5], [9], air quality forecasting [6], wind speed forecasting [7], etc. Safat *et al.* [5] conducted an experiment employing all of the models that can predict more than 35 types of crime and provided a yearlong crime prediction. Feng *et al.* [1] did such a study using prophet model, neural network, and LSTM along with data visualization. This model works better only with the data from the previous three years. Sometimes, the LSTM arrangement makes it impossible for the memory cell to retain information over numerous time steps [8]. However, the model of Feng *et al.* needs more training data and data mining techniques to understand crime patterns better. Data mining is one of the fundamental techniques of BDA. It is innovative and growing research and helps deduce useful information and hidden patterns from data. It not only helps us in discovering new knowledge but also in enhancing our mastery of known ones [2]. Rayhan *et al.* [9] performed a Spatio-temporal Attention-based study to predict crime of top 4 categories. Their model cannot predict categories having a small amount of data.

Since the prediction of crime can assist authorities in maintaining social order, a deep learning-based multi-module generic method for predicting crime in various cities is presented. This study forecasts the number of crimes in various categories in several districts for each city. It also analyzes how the trends impact crime occurrence. Locations were considered to picture the crime hot spots better. We processed the characteristics to obtain the required form and then chose the characteristics based on the correlation. After that, our proposed model was trained and evaluated. There are three types of information in the dataset: categorical, temporal, and spatial. We developed the Attention-LSTM (ATTN-LSTM) model to process the categorical-temporal data and the Stacked Bidirectional LSTM (St-Bi-LSTM) model to process the spatial information. Predicting crime with the same model for multiple cities may result in a significant loss since the attributes of one location may not have the same quantity of unique data as another. Hence, feature level fusion (FLF) and decision level fusion (DLF) modules were applied to overcome all the drawbacks of the current state-of-the-art. Our goal throughout the study was to predict crime more effectively than previous methodologies employing time and particular location. Figure 1 shows an abstract depiction of this model. Our contributions to this work are the following:

- We proposed a robust multi-module approach for dealing with category, geographical, and temporal information. A method for combining these characteristics into a single model utilizing two degrees of fusion was developed.
- LSTM cells were fabricated with Swish activation to deal with the vanishing gradient issue and negative inputs in backward propagation to emphasize the utilization of LSTM.

- A weighted system was implemented to determine the loss for the DLF module. The calculation is based on the inputs of this module.
- A method for combining three models was devised in DLF. We utilized the majority vote approach and all-pairs shortest distance to forecast each crime category based on the outputs of the three models. Hence, the suggested model predicted crime with minimal inaccuracy.

Our code is uploaded at https://github.com/NowshinTasnim/Spatio_Temporal_Crime_Prediction.git.

The remaining portions of this paper are organized as follows. We reviewed our study on similar works in Section II and analyzed the data in Section III. The motivation of this work is provided in Section IV. An explanation of the developed architecture is given in Section V, and the working procedure for this study is discussed in Section VI. In Section VII, the results of the suggested work are analyzed. Finally, the paper is concluded in Section VIII.

II. LITERATURE REVIEW

With the ever-changing society, crime patterns are also changing. Moreover, the incidence of crime is increasing. The traditional approaches are mostly out of date in this regard. Many deep learning-based supervised, semi-supervised, and unsupervised techniques are used to predict crimes and check the trends. The integration of modern technology into crime prediction helps the authorities take necessary precautions.

A. BASELINE NETWORKS

Feed-forward ANN or multi-layer perceptron is most successful in time-series forecasting as it does not require the data distribution beforehand [10]. Using DL in time-series forecasting, the temporal dependence and structure can be easily learned [11]. Integrating Recurrent neural network (RNN) configuration in deep learning has changed the way of processing data in the case of forward-dependency networks. Moreover, this configuration can solve many real-world problems. LSTM can process an entire sequence, not only a single data point. So, while working with time-series data, the LSTM network is quite useful [5], [12]–[14]. Schuster *et al.* [15] used the concept of LSTM and improved it by coming up with the idea of Bidirectional LSTM (Bi-LSTM). The model can better understand the context by including feed-forward and feed-backward networks. Said *et al.* [16] and Kim *et al.* [17] showed that the use of Bi-LSTM enables one to get the correlations and changed values of variables in the series simultaneously during the processing of multivariate time series data. The variation of each time-series data plays an important role in forecasting. Using Bi-LSTM, we can analyze these variations.

Said *et al.* [16] described the use of stacking several Bi-LSTM layers in multivariate time-series data in case of prediction. By using such layers, the model can learn about spatial-temporal features from the dataset and predict the next timestep more accurately [18], [19]. Sometimes, these stacked layers reduce unnecessary information while processing the series of data. St-Bi-LSTM performed better in predicting the future from the sequence instead of using several LSTM layers [20]. So far, Bi-LSTM has performed outstandingly in diverse fields that include time-series-based forecast, COVID-19 case prediction, wind power forecast, and many more [16], [17], [21].

A more appropriate way to reduce redundant data while processing is to use an Attention-based model. It takes the relevant parts of the input data and removes the redundant data while performing a task. An attention-based model works well for sequence modeling as the distance between the input and output layers does not affect the modeling dependencies [22], [23]. In most contexts, this type of mechanism is used with a RNN to create an encoder-decoder based sequence to sequence architecture [22], [24]–[28]. Though all the above mechanisms work smoothly for a small sequence, they are not effective enough for a long one. In such circumstances, the self-attention or intra-attention mechanism works better. This method creates a relationship between several positions of a single data sequence to represent this single input. This representation helps in maintaining long-range dependency [29]. The path length of long-range dependency is generally the shortest to learn about the sequence easily [30]. Moreover, the execution of self-attention layers is faster than the RNN in most cases [29]. Attention and Self-attention mechanisms are mostly used for computer vision and natural language processing like speech enhancement, text prediction, and summarising [25], [31]–[34]. Nevertheless, only one approach is not enough for dependent data of varied features. It requires the fusion of models [35], [36].

B. CRIME PREDICTION AND CLASSIFICATION

BDA has shown tremendous results in criminological aspects in finding the trends and relationships between data [1]. Feng *et al.* [1] described how the stateful LSTM and Prophet models work in crime analysis. They tracked the crimes and predicted likelihood by following related facts and patterns. Their model works better with three years of data but cannot perform similar results for data spanning a longer period of time.

A Spatio-temporal based RNN was introduced by Wang *et al.* [37], [38] to predict the total number of crimes. This method used a multi-factor crime prediction model consisting of adaptive hierarchical structured residual convolution units and non-convolution models. The layers are independent of each other in this model. They achieved a good accuracy but could not detect categories separately. They also used the internalization technique to address the resource consumption issue for its deployment in the real world [37]. However, the spatial data mining model proved to be an excellent approach to detect crime hot spots [39].

Agarwal *et al.* [40] and Tayal *et al.* [41] used the K-means clustering-based model to show the crime patterns based on year. It also has the same problem as Spatio based system. These clustering models can work only with categories

TABLE 1. Brief summary of the related works.

Authors	Techniques	Crime Dataset	Weakness/Remarks
Feng et al. [1]	LSTM and Prophet model	San Francisco, Chicago, and Philadelphia	Cannot perform well for data more than three years.
Feng et al. [2]	KNN, Naive Bayes, Random forest, XGBoost	San Francisco, Chicago, and Philadelphia	Did not correlate with time-based trends.
Rayhan et al. [9]	AIST	Chicago,USA	Cannot work with small dataset.
Wang et al. [37], [38]	Spatio-temporal based RNN	Los Angeles, USA	Cannot detect categories separately.
Agarwal et al. [40]	Kmeans clustering	England and Wales, UK	Cannot work with small dataset.
Tayal et al. [41]	Kmeans clustering	India	Cannot perform time-based analysis.
Kumar et al. [42]	Naive-Bayes-based model	Cheltenham, UK	Trained with specific categories.
ToppiReddy et al. [43]	KNN and Naïve Bayes classification	UK	Low accuracy across various categories.
Sivaranjani et al. [44]	K-means, Agglomerative, and DBSCAN	Tamilnadu, India	Calculate the exact time of the crime is not practical.
Pednekar et al. [45]	K-means, Agglomerative, and DBSCAN	India	Calculate the exact time of the crime is not practical.

TABLE 2. Attributes in dataset.

Crime Data	Attributes
San Francisco	PdId, IncidntNum, Incident Code, Category, Descript, DayOfWeek, Time, PdDistrict, Date, Resolution, Address, X, Y, location, Neighborhoods information, Supervisor Districts, Fire Prevention Districts, Zip Codes, Fix It Zones, Civic Center Harm Reduction Project Boundary, CBD, BID and GBD Boundaries as of 2017, Areas of Vulnerability, Central Market Tenderloin Boundary, HSOC Zones, OWED PublicSpaces, datetime.
Chicago	ID, Case Number, datetime, Block, IUCR, Primary Type, Description, Location Description, Arrest, Domestic, Beat, District, Ward, Community Area, FBI Code, X Coordinate, Y Coordinate, Year, Updated On, Latitude, Longitude, Location

having more than a certain amount of data and cannot perform time-based analysis.

Rayhan *et al.* [9] developed an attention-based deep learning model capturing the non-linear spatial dependence and temporal patterns of a particular crime category. The self-attention network can emphasize the dependencies among the features and get better results than uniform LSTM in predicting the time-series-based data. They kept the model's fundamental structure interpretable. This model dynamically establishes a spatial-temporal association for each crime category based on prior crime occurrences and repeating crime trends. The limitation of this work is that the crime categories must have a large amount of training data.

Kumar *et al.* [42] proposed a Naive-Bayes-based model for crime classification. Their method involves combining the history of some crime occurrences with incident-level crime data to identify the most likely criminal in a given incident. It works well for some categories, but the cumulative accuracy was near 50%. A similar kind of approach was taken by ToppiReddy *et al.* [43] for crime classification. They used the KNN and Naïve Bayes classification systems. They took the day information with the location to know whether a crime would occur and in which category the crime would occur.

An improved method of crime classification using Clustering approaches was proposed by Sivaranjani *et al.* [44] and Pednekar *et al.* [45]. They used the K-means, Agglomerative, and DBSCAN clustering techniques to make crime clusters. Then they merged the information from the three resultant clusters to predict the class. This approach has high accuracy, but calculating the exact time of the crime is an impossible job.

Gradient boosted decision trees worked better than KNN, Naive Bayes, and Random Forest Classification for correlation-based selected features and categories with a large number of data points. Feng *et al.* [2] used forecasting methods to generate data using crime trends of recent years. This model predicts crime categories for a given time and location using tree classification. It performs better than KNN and Naive Bayesian approaches. They merged the crime categories having a small amount of data. However, for a relationship with the trends, time-based analysis is necessary.

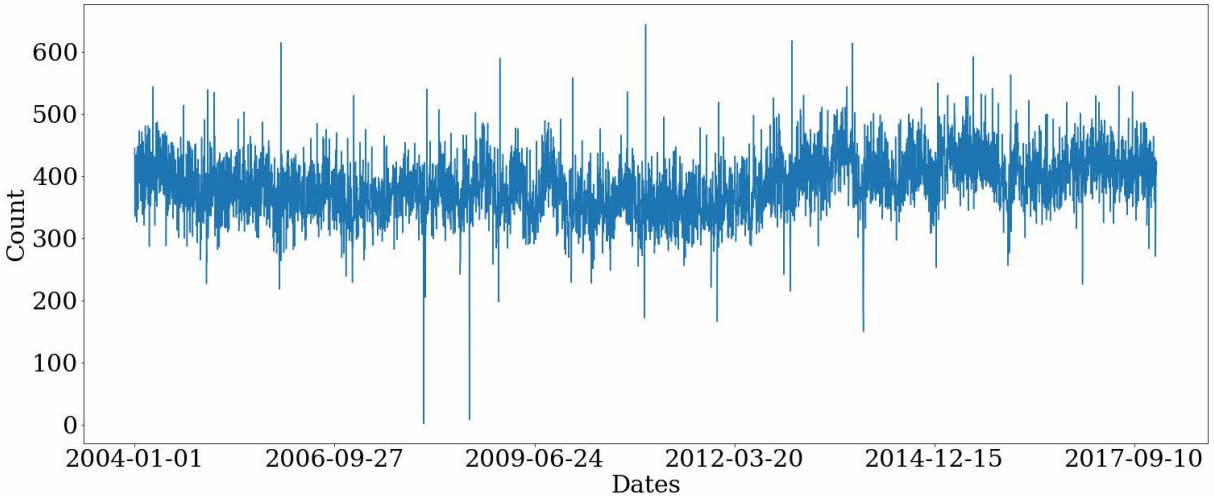
Different deep learning models perform better in a given sector for a specific criterion. However, for crime forecasting, the success rate of those models has not reached equilibrium. A summary of the current state-of-art is presented in Table 1. There is still work to be done to address the deficiencies of today's state-of-the-art.

III. EXPLORATORY DATA ANALYSIS

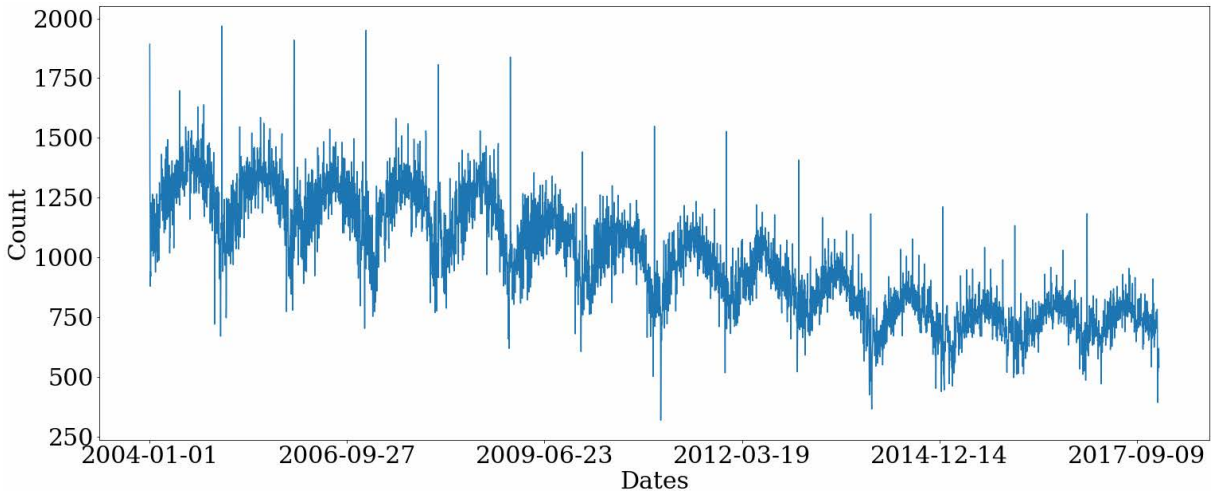
A. DATASET COLLECTION AND DESCRIPTION

We collected the crime data of San Francisco and Chicago, respectively, from the San Francisco city-county data portal [46], and the Chicago data portal [47] from 2003 to 2017.

There are 2, 115, 112 crime incidents in San Francisco and 5, 547, 827 crime incidents in Chicago dataset. However, we are using data from 2004 to 2017 due to a data deficiency in 2003. Hence, 1, 970, 039 crime incidents for San Francisco



(a) Total count over time for San Francisco dataset



(b) Total count over time for Chicago dataset

FIGURE 2. **Total crime in each day over the 14 years.**

and 5, 071, 866 crime incidents for Chicago are in our dataset. The existing attributes are listed in Table 2.

The daily crime occurrence from 2004 to 2017 for San Francisco and Chicago are shown in Fig. 2. From the figures, we observe that the two cities' crime rate changes are not the same. Moreover, the Chicago dataset shows seasonality. The crime rate in San Francisco has been almost the same over the 14 years. However, it is decreasing over the 14 years in Chicago, which is still higher than in San Francisco. Seasonal trends in the data impact the whole model in time-series forecasting. In the case of crime prediction, these kinds of trends help to understand the pattern and find out the crime hot spots.

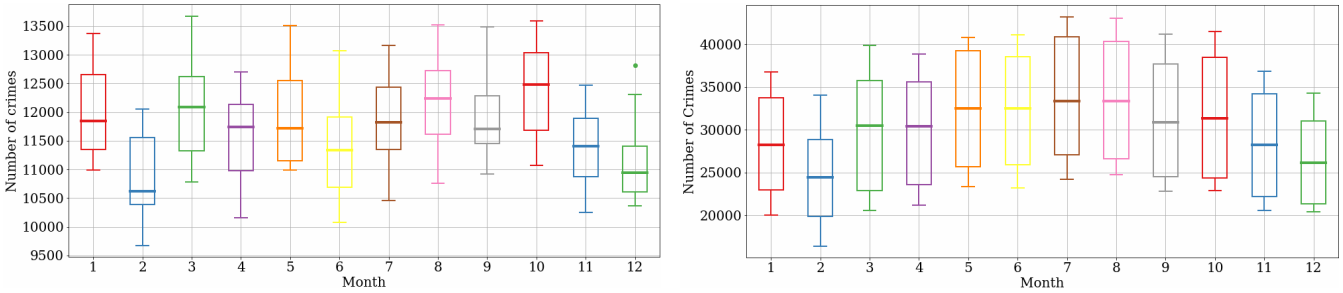
To anticipate crime, we need information from prior instances, such as the time and location of the crime and the sort of crime committed. So, for each city, the following existing features from the dataset were used:

1) Date – Date of the crime occurrence.
2) Time – Timestamp of the crime incident.
3) Categories/Primary type – Type of the crime that took place.
4) PdDistrict/District – Police Department District name where the crime took place.

The weather data of San Francisco and Chicago were taken from Wonder Weather Forecast [48] and used to check the temperature trends on the Kelvin scale. The publicly available dataset consists of hourly-based weather information. Hence, we utilized the average temperature of each hour from this dataset.

B. ANALYZING THE EXISTING FEATURES

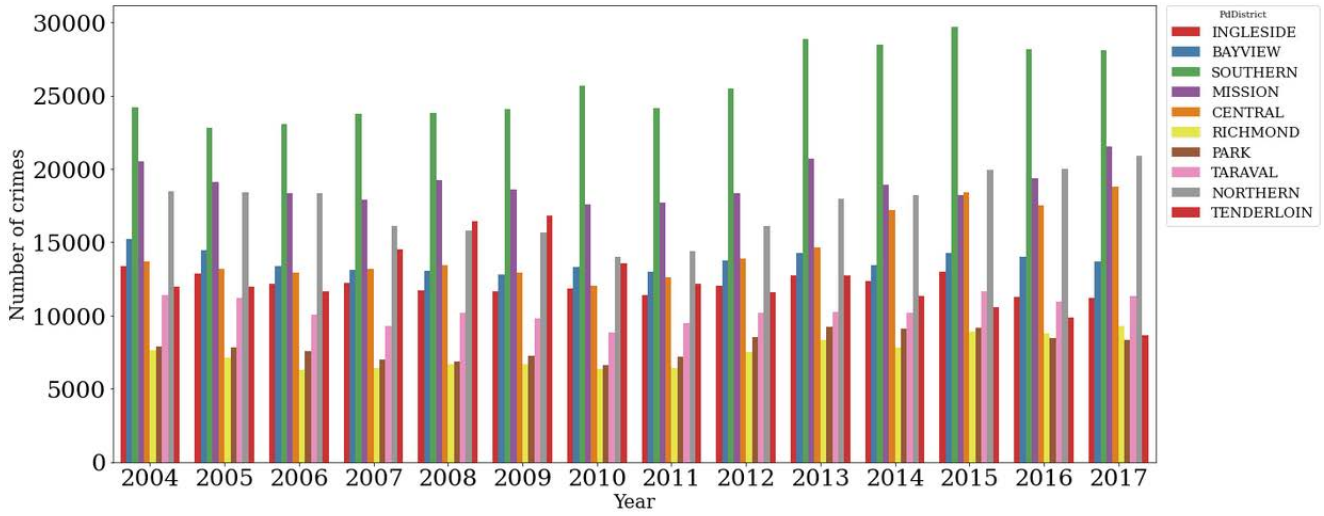
After checking the correlation between features, we chose features having a higher correlation from the acquired dataset. The information about year and month from the dates column of the San Francisco dataset and the datetime column of the Chicago dataset was retrieved. We visualized the monthly distribution of crimes during the 14 years using these extracted data, as shown in Fig. 3. From Fig. 3a, crime data



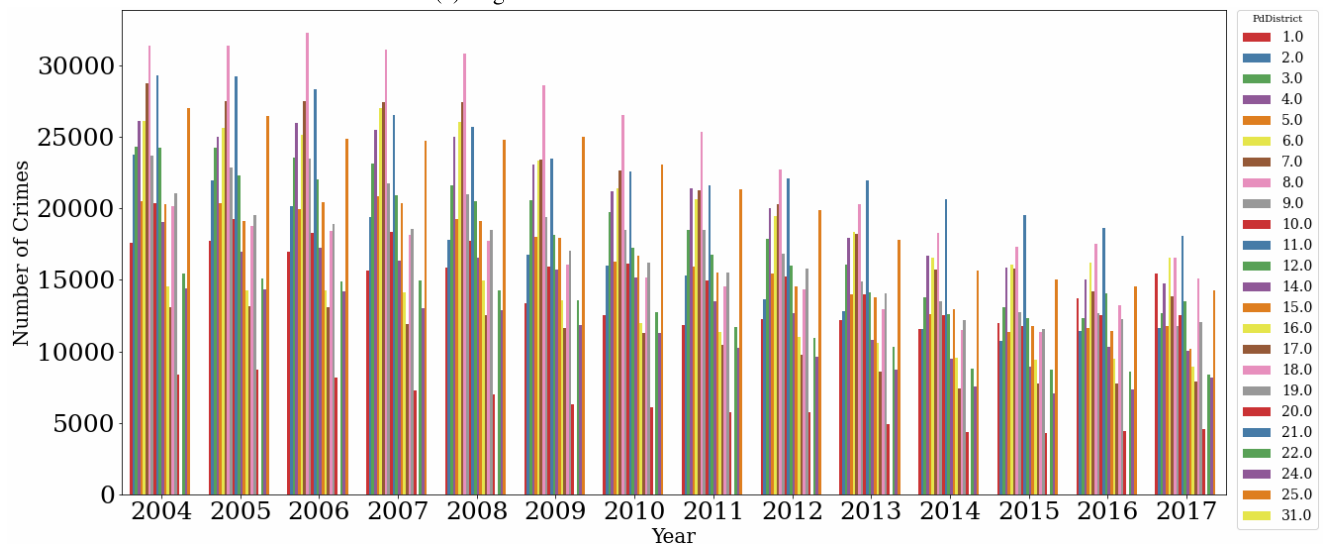
(a) Monthly total crime data of San Francisco

(b) Monthly total crime data of Chicago

FIGURE 3. Monthly based data distribution over the 14 years.



(a) Regional crime distribution of San Francisco



(b) Regional crime distribution of Chicago

FIGURE 4. Yearly number of crimes in each police department.

distribution can be observed. It is high for January, March, May, August-October over the 14 years for San Francisco. In May, July, August, and October, the crime rate is high in Chicago, as shown in Fig. 3b. Generally, in San Francisco, the temperature is high from August to October due to the summer season.

For the same reason, the temperature in Chicago is high in July and August. Hence, there is a possibility that temperature

plays a role in crime occurrence. We can also see that the crime rate in San Francisco is high in March and May. The crime rate in Chicago is high in May and October, but the temperatures in these months are lower than in the summer season.

Analyzing the yearly number of crimes per police district for San Francisco and Chicago from Fig. 4, we can find the most crime occurring districts. Throughout the 14 years, the maximum number of crimes occurred in the Southern District of San Francisco and the district 8 and 11 of Chicago. These were the hot spots for crime in those years. With those facts, the deduction is that the environment of a district or area holds importance in crime occurrence. Thus, while predicting the crime occurrence, considering the districts as one of the critical features for the model is necessary. However, many of the existing models did not consider the space in the feature.

IV. TECHNICAL MOTIVATION

After reviewing the data, inevitable flaws are discovered in the current models. These flaws aspired us to establish a generalized model that will perform better than the current state-of-arts. To do so, we investigated the shortcomings of these models and presented several improvements.

A. CHUNKING OFF THE UNNECESSARY INSTANCES AND EMPHASIZE ON RELATED INSTANCES

Feng *et al.* [1], [2] trained their model using the whole city as a location. However, designating the region of a city plays an essential function in controlling crime incidence. In our work, the input features include the information about the locations utilizing the police department district for each city. We needed to implement Bi-LSTM and ATTN-LSTM in the model to deal more efficiently with the area-based information.

Our model has a St-Bi-LSTM layer to train the information of the police departments of the cities. It also has an Attention-based sub-model to learn the temporal aspects of the cities. Bidirectional RNN has two layers side-by-side. The second layer is a replica of the network's first recurrent layer. In the first layer, the input sequence is the provided input. The input sequence in the second layer is the reverse of the provided input. Even if the input sequence is very long, the chances of losing any information from the whole input become pretty minimal here. For RNN, the depth of the network is more important than the memory cells of a layer. The depth of our model is boosted by stacking two Bi-LSTM layers. By chunking off some unnecessary observed instances of the first layer in the second layer, this model helps the network to predict better than the network having one Bi-LSTM layer with the same number of memory cells. Bi-LSTM also solves the vanishing and exploding gradient problem that vanilla RNN has. Furthermore, it provides considerably cleaner back-propagation compared to vanilla RNN.

The attention model overcomes the limitations of encoder-decoder-based approaches. This model examines the relationships between the nodes and maintains the nodes that play a significant role in creating the output. As a result, the model selects the appropriate nodes for training and minimizes the input size of the next layer [29]. It is important to keep only the relevant instances in sequential data training and remove unnecessary observed instances to produce a better output.

B. ACCELERATING CONVERGENCE IN LEARNING PROCESS USING TRANSFER LEARNING

Rayhan *et al.* [9], Feng *et al.* [1], [2], Wang *et al.* [37], [38] made remarkable progress in crime analysis and forecasting. However, their models require a longer compilation time for the large datasets. Furthermore, if the dataset contains data spanning more than a decade, these models will not perform well. Each city in our collection had a massive quantity of data. To address the compilation time issue, employing the transfer learning technique to converge a learning process is a smart option. In the case of solving a new problem, a model can use the previously trained model knowledge if the two are comparable. This action is called "Transfer learning". If there is not a decent amount of data to train with or the training time is excessive, transfer learning technology can help solve these problems. It makes the learning process faster and increases accuracy [49].

For example, D_{Source} and D_{Target} are two different domains having learning tasks T_{Source} and T_{Target}, respectively, where $T_{Source} \neq T_{Target}$. However, the knowledge of the source model is similar to the target model's. Hence, one can use this knowledge of the first model for the second model by transferring it (Fig. 5). Here, both of the models produce different outputs. In this work, the temporal features of the Chicago dataset are learned by applying the knowledge from the ATTN-LSTM model, which has only the temporal features of San Francisco. It reduced the training time.

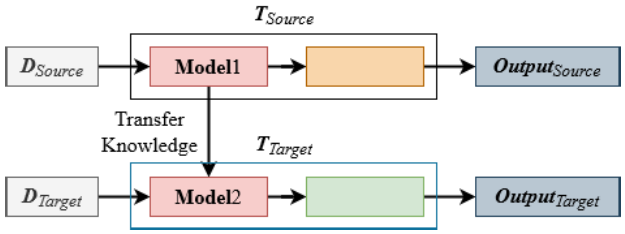


FIGURE 5. Transfer learning technique.

C. GENERALIZING THE MODEL

Some developed models used FLF to get generalized performance. However, to get a better result, DLF is also required [2], [9], [38]. Our model employed two levels of fusion. The Fusion approach can transform weak learners who are marginally better than a random guess into strong, aggregated learners who can make correct predictions. Furthermore, the generalization power is far greater than that of base learners [50]. In our work, the proposed model is a generalized model that can predict crime in any location without losing its efficacy. Different cities have different values for a single feature. Our model deals with these various

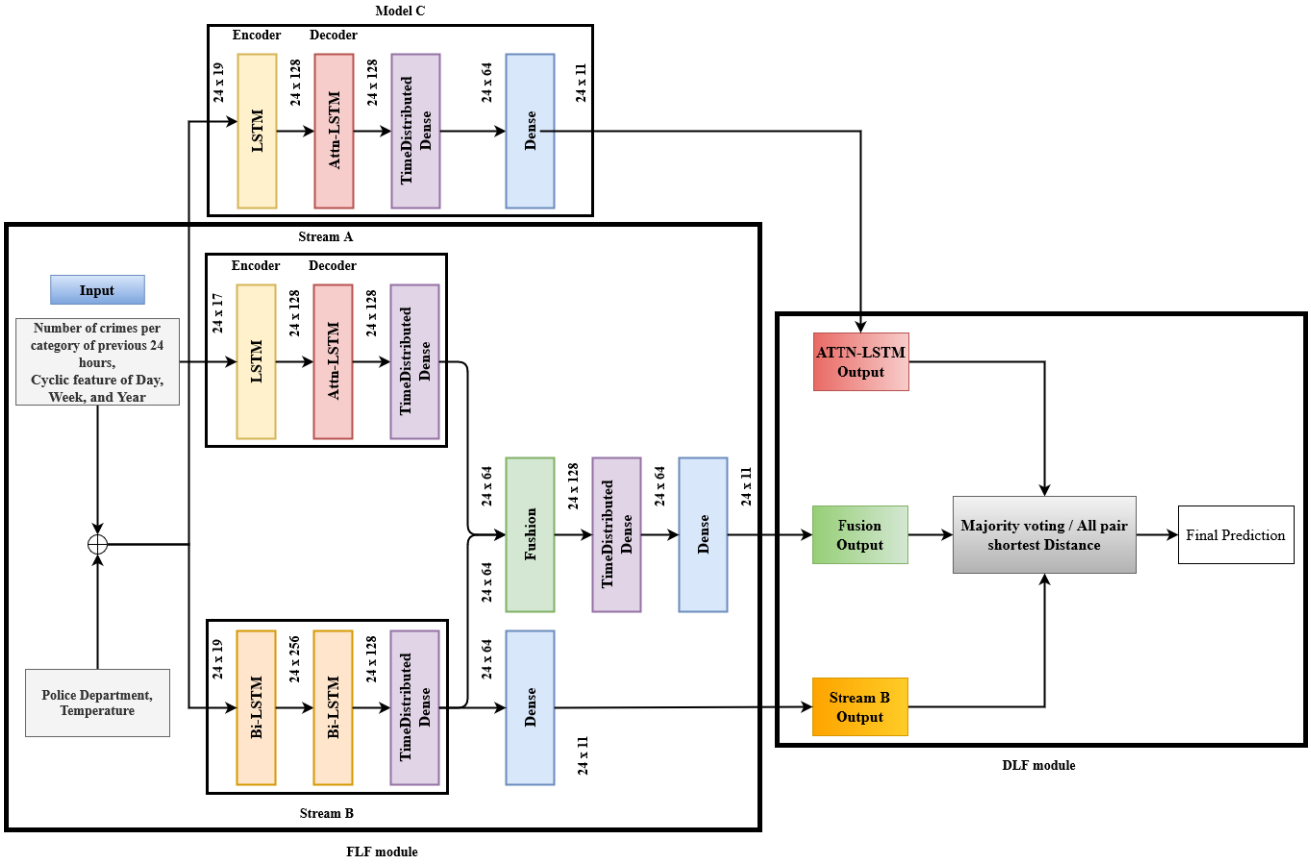


FIGURE 6. Attention and sequential fusion based proposed architecture.

values by using FLF. It also has a DLF module to get a better prediction result.

V. PROPOSED ARCHITECTURE

DL approaches work phenomenally for time-series-based forecasting. Taking inspiration from our study, we developed a DL-based architecture for predicting crime. Fig. 6 depicts the proposed architecture. The whole architecture is divided into four sub-models. The proposed work uses the St-Bi-LSTM model, the ATTN-LSTM model, and two levels of Fusion models. This novel architecture can overcome the issues of the current state-of-art.

In our work, the LSTM cells of the Bi-LSTM layers and ATTN-LSTM layers are designed using the Swish activation function to avoid particular concerns with our dataset.

1) LSTM CELL

LSTM cells are a special kind of RNN cell with the capability of handling long-term dependencies. It has three gates- forget gate (f_t), input gate(i_t), and output gate(o_t). The designed LSTM cell looks like Fig. 7. The equations are the following:

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ Swish \end{pmatrix} * W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

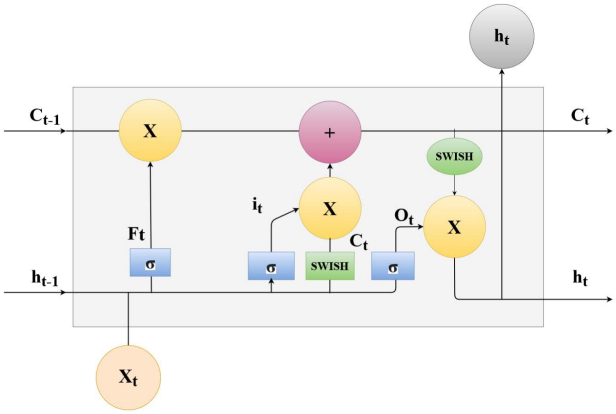


FIGURE 7. LSTM cell.

$$\begin{aligned} c_t &= f \oplus c_{t-1} + i \oplus g \\ h_t &= o \oplus Swish(c_t) \end{aligned} \tag{1}$$

2) SWISH ACTIVATION

Swish is a smooth, non-monolithic function that matches or outperforms ReLU in different machine learning problems. It is derived from SILU activation. The equation for Swish is

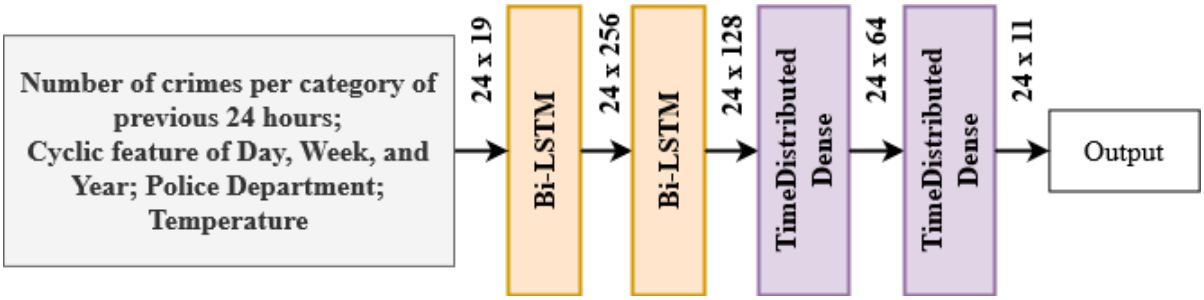


FIGURE 8. St-Bi-LSTM.

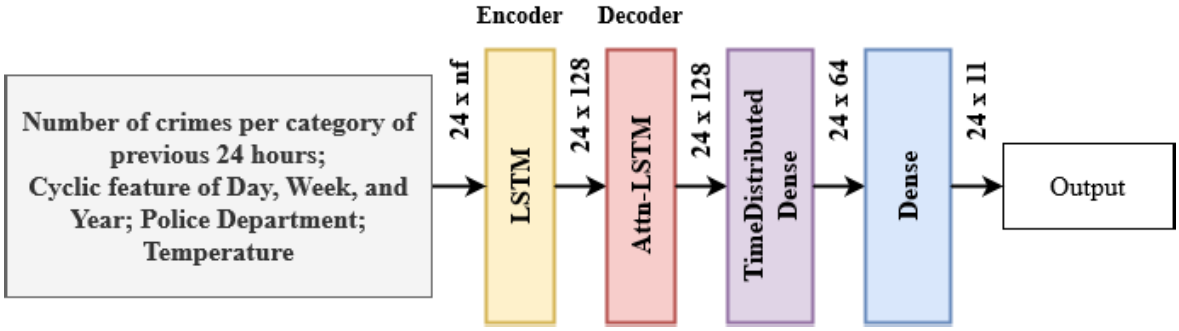


FIGURE 9. ATTN-LSTM.

the following:

$$Swish(v) = \frac{v}{1 + e^{-\beta v}} \tag{2}$$

where, β is a trainable parameter. Swish can be termed as v time *sigmoid* βv, and this function does not have a vanishing gradient problem. Also, ReLU produces 0 output for negative inputs. It cannot be back-propagated where Swish can partially handle this problem.

The description of the sub-models is given in the following.

A. STACKED BIDIRECTIONAL LONG SHORT-TERM MEMORY (ST-BI-LSTM)

The St-Bi-LSTM model is designed with two TimeDistributed wrapped Dense layers for training the geographical, temporal, and derived features of the cities.

A TimeDistributed wrapper predicts one value per timestep for the whole input sequence. It allows applying the mentioned layer to each part of a sequence. So, this requires that the Bi-LSTM hidden layer return a sequence of values (one per timestep) rather than a single value for the whole input sequence. Two Dense layers with a TimeDistributed Wrapper minimize the size of the input of those layers and consider each part of the sequence while doing so. Finally, we got the desired sequence of outputs for this model. The architecture of this model can be seen in Fig. 8. Some dominant parameters for this model are given in Table 3.

TABLE 3. St-Bi-LSTM model parameters.

Hyper Parameter	Value
Number of features	19
Number of units per layer	Bi-LSTM - 128
	Bi-LSTM - 64
	TimeDistributedDense - 64
	TimeDistributedDense - 11
Early stopping	Based on Validation loss
	Patience = 10
	Restores best weights
Optimizer	Adam
Activation	Swish

B. ATTENTION BASED LONG SHORT-TERM MEMORY (ATTN-LSTM)

In Fig. 9, the architecture for an encoder-decoder model with ATTN-LSTM is shown. Here, *nf* indicates the number of features. The value of *nf* is 19 for Spatio-temporal features and 17 for the architecture with only the categorical-temporal feature. At first, the features are given as input to an LSTM layer, which serves as an encoder.

Then the encoded output is passed to the multiplicative self Attn-LSTM layer to decode the data sequence. A multiplicative self-sequential-attention layer performs better for sequential data input than a vanilla attention layer. First, the input sequence is taken and matched as a row and column of a matrix. Then the hidden states are calculated. The hidden state vectors (h) are the sequence of specific features of the

input. After that, the context vector (l) is computed using the weighted sum of the h vectors. The attention vector (e) gives the output score of the feed-forward neural network. Applying Softmax to calculate the weights (a), the scores of the attention vector will be distributed fairly. The equations for the multiplicative self-attention layer are following:

$$a_t = Softmax(e_t) \tag{3}$$
$$e_{t,t'} = Swish(x_t^T W_a x_{t'} + b_a) \tag{4}$$
$$l_t = \sum_{t'} a_{t,t'} x_{t'} \tag{5}$$
$$h_{t,t'} = Swish(x_t^T W_t + x_{t'}^T W_x + b_t) \tag{6}$$

where, a holds the attention weights, e is the attention vector, l is the context vector, and h holds the scores.

The decoded output is passed to a dense layer with a TimeDistributed wrapper so that each part of the sequence is processed separately. This layer is followed by another dense layer to downstream the output sequence. The main parameters for this model are given in Table 4.

TABLE 4. ATTN-LSTM model parameters.

Hyper Parameter	Value
Number of features	19 (For Spatio-temporal analysis),
	17 (For Temporal analysis)
Number of units per layer	LSTM - 128
	Self-Attention - 128
	TimeDistributedDense - 64
	Dense - 11
Early stopping	Based on Validation loss
	Patience = 10
	Restores best weights
Optimizer	Adam
Activation	Swish

C. FEATURE LEVEL FUSION (FLF) MODULE

The Fusion sub-model concatenates the ATTN-LSTM (where $nf = 17$) and St-Bi-LSTM model. This architecture is shown in Fig. 6 as the FLF module. ATTN-LSTM and St-Bi-LSTM model layers (excluding the last layer of both models) is used as stream A and stream B, respectively, for merging. The weights of these two streams are not updated during the training. Stream A contributes to temporal features of a city, and stream B contributes to spatial features. These streams help to create a generalized model for forecasting crimes. Then Stream A and Stream B are concatenated, followed by a TimeDistributed dense layer and a dense layer. This kind of fusion is known as ''Feature level Fusion''.

D. DECISION LEVEL FUSION (DLF) MODULE

To avoid problems due to anomalies in the data and to get more accurate results, the DLF module is introduced. Here, the ATTN-LSTM (for Spatio-temporal features), St-Bi-LSTM, and FLF are all taken to get the final prediction. This sub-model is shown in Fig. 6 as the DLF module. DLF chooses the most appropriate outcome among the three outputs. For the DLF module, majority voting and the all-pairs shortest distance are implemented to find our proposed model's final crime prediction. At first, the module checks if the predictions for each crime category support the majority voting system. Otherwise, the all-pairs shortest distance is applied for selecting the output. This module uses a weighted average system to measure the loss. The following is a short description of the prediction and loss calculation techniques in the DLF module.

1) MAJORITY VOTE

In the case of a majority vote, if an element frequently occurs and more than the rest of the inputs, it is a majority element. In our work, if the predictions of a category for any two models are the same, it is the final output. The equation for the majority vote is:

$$C_c^* = \sum_{i=1}^{m} \sum_{j=1}^{m} (\hat{y_{ic}} == \hat{y_{jc}}) \tag{7}$$

where, m denotes the number of models; $\hat{y}$ denotes the prediction of a specific hour for those models; c is the crime category; C^* is the majority vote for the c crime.

2) ALL-PAIRS SHORTEST DISTANCE

Finding the minimum distance between any two elements is the intention of the all-pairs shortest distance technique. Sometimes, to calculate the shortest distance, the distance from the mean of the elements is used [51]. Our DLF model uses the deviation from the mean to find the final prediction for each category among the 3 outcomes. The equation is:

$$D_c = \min_{i=1}^{m} \left(\frac{1}{m} \sum_{j=1}^{m} (|\hat{y_{ic}} - \hat{y_{jc}}|) \right) \tag{8}$$

3) WEIGHTED AVERAGE LOSS

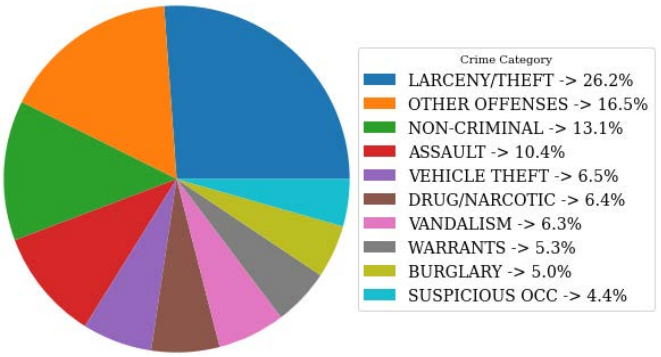
This technique prioritizes some values by assigning weights. In this work, the DLF module calculates the average loss for each city using the weighted average loss for each district of that city. It calculates the weight for each model by checking the proximity of the prediction to each category. The equations for calculating loss are:

$$loss_{w_L} = \frac{1}{m} \sum_{j=1}^{m} (weight_{jL} * Loss(\hat{y_{jL}})) \tag{9}$$

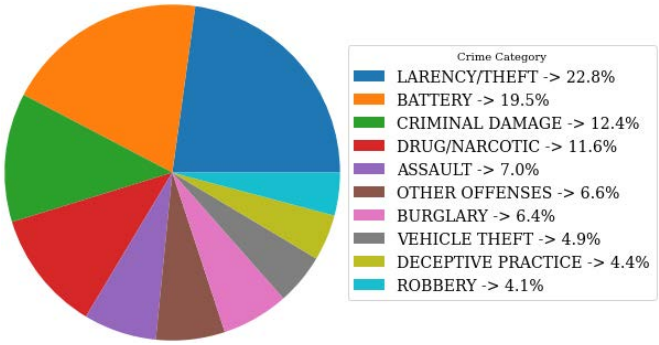
where, L is the PdDistrict.

$$loss_w = \frac{1}{p} \sum_{L=1}^{p} (weighted\_avg\_loss_L) \tag{10}$$

where, p denotes the number of total PdDistricts for a city.



(a) Percentage of some crime categories for San Francisco



(b) Percentage of some crime categories for Chicago

FIGURE 10. Top 10 crimes over the 14 years.

VI. EXPERIMENT

A. EXPERIMENTAL SETUP

For execution purposes, we used the google colab platform for this work. The specifications of the Google Colab platform are:

- 1xTesla K80 (2496 CUDA cores)
- 1xsingle core hyperthreaded Xeon Processors @2.3Ghz
- 13 GB RAM
- 108 GB Run time HDD
- OS: Linux Kernel

Short experiments like validating the code's functionality were performed on a desktop computer.

B. FEATURE SELECTION METHOD

Features play a vital role in predictive models. The model predicts more accurately when the correlation between features and the target value is strong. In this study, the R-value is used to measure this correlation. The equation for R-value is given below:

$$r = \frac{\sum_{i=0}^{ns}(f_i - \bar{f})(t_i - \bar{t})}{\sqrt{\sum_{i=0}^{ns}(f_i - \bar{f})^2 \sum_{i=0}^{ns}(t_i - \bar{t})^2}} \tag{11}$$

where, r is the correlation coefficient, f_i is the values of the feature variable in a sample, $\bar{f}$ is the mean of the values of the feature variable, t_i is the values of the target variable in a sample, $\bar{t}$ is the mean of the values of the target variable, and ns is the total number of samples in the dataset.

C. DATA PRE-PROCESSING

The data were not in the form we expected. Hence, we processed the data to get the required formation. The steps of pre-processing are given in the following.

A sorted, hourly-based DateTime column was made for all the data by merging the Date and Time columns. Information about the day, year, and week was extracted from the date. Eq. (12) and (13) encoded this information into a signal. The equations for cyclic features are the following:

$$Cyclic_x = sin(timestamp * (2 * \frac{\pi}{s})) \tag{12}$$

$$Cyclic_y = cos(timestamp * (2 * \frac{\pi}{s})) \tag{13}$$

where,

$s = 86400$ seconds for a day,
$s = 604800$ seconds for a week, and
$s = 220898664$ seconds for a year

These signals help to co-relate the periodical nature with the data. After this step, the features of missing timestamps within the chosen 14 years range were masked using the ones of the first timestamp of the dataset.

From the weather data, we took the average temperature (Kelvin) per hour. It was divided into 3 categories according to (14). Here, 0 indicates low, 1 indicates medium, and 2 indicates high. This categorized temperature information is the derived feature for our hourly timestamp-based crime data.

$$temp = \begin{cases} 0, & \text{if } temp_{avg} \leq 273 \\ 1, & \text{if } 273 < temp_{avg} \leq 305 \\ 2, & \text{otherwise} \end{cases} \tag{14}$$

There are different unique values for the two features-Category and District. For these two cities, some of the data pre-processing steps are different. These steps are described below with respect to cities:

1) SAN FRANCISCO CRIME DATA

The total number of crimes per category was analyzed to find the 10 top ones out of 38 crimes which are shown in Fig. 10a. The results showed that "Larceny / Theft" occurred more often than other crimes. The second highest crime occurrence was "Other offenses". The third-highest one was "Noncriminal". During the analysis, it was found that the lowest total number of crimes for a category was 14. Many categories had less than 100, 000, except the top 7. For balancing the dataset, these low count categories were merged. There were 38 categories in total. Each category was mapped with a unique number from 0 to 37. We divided the categories with fewer data into 3 groups for merging, leaving the top 7 categories. Clustering approaches were used to make these groups - GRP0, GRP1, and GRP2. GRP0 consists of warrants, burglary; GRP1 consists of suspicious occ, robbery, missing person, fraud; and GRP2 consists

of the rest categories. After merging, there were 10 types of crime categories. These Categories' name and mapped ids are: larceny/theft (1), other offenses (2), non-criminal (3), assault (4), vehicle theft (5), drug/narcotic (6), vandalism (7), GRP0, GRP1, and GRP2. After that, the unique PdDistricts name was converted to a unique numerical value as PdId ranging from 1 to 10. Then, we counted each category's crimes based on hourly timestamp and police department district id.

2) CHICAGO CRIME DATA

After analyzing the total number of crimes per category, the 10 top ones out of 31 crimes are acquired. These crimes are shown in Fig. 10b. The results showed that ''Larceny / Theft'' occurred more often than other crimes. The second-highest crime occurrence was ''Battery''. The third-highest one was ''Criminal damage''. In the case of the Chicago dataset, the lowest total number of crimes for a category is 11, and there are many categories having data less than 300, 000, except the top 7. We merged these categories to have balanced data. The 31 categories in this dataset were mapped with a unique number from 0 to 30. The smaller categories were merged into 3 groups, the same as San Francisco data. The GRP0 consists of deceptive practices and vehicle theft. The GRP1 consists of robbery and criminal trespass, and GRP2 consists of the rest categories. Therefore, the 10 categories name and their mapped ids are larceny/theft (1), battery (2), criminal damage (3), drug/narcotic (4), assault (5), other offenses (6), burglary (7), GRP0, GRP1, and GRP2.

So, the acquired data from both datasets are the following:

1) PdDistrict id – Unique id given for unique police department districts.
2) Cyclic encoded the day, week, and year information data.
3) Count of crime per category for the hourly timestamp.
4) Categorized temperature for the hourly timestamp.

Among these four features, PdDistrict id and temperature differ from city to city. Hence, we excluded these two features from the transfer learning technique to create a generalized model. Since these two features are among the main factors in crime, these features are processed using another model. The correlation between acquired features and the crime category is shown in Table 5 using (11).

D. WINDOW GENERATION

The acquired data was split into train, test, and validation set in a ratio of 7:1:2. Since our dataset is time-series-based, the first 70% data are for training, the last 10% data are for testing, and the rest of the 20% is for the validation. Then windows of data having 24×17 inputs and 24×19 inputs for each window were generated. The windows mainly represent the input sequences. The 24 denotes the data of 24 hours of a day. We mapped the data of 24 sequential hours per sequence for input. The model predicted the same data sequence length after moving one hour ahead from the starting of the input sequence. Thus, the model is taking 24 hours data as an input data point and is predicting data of the next hour and the previous 23 hours as shown in Fig. 11. In 24×19, the 19 denotes the 19 features to predict in the next time step. These features are the encoded signals of day, week, and year, police department district id, temperature category, previous hour's count of 10 categories, and the total number of crimes in the last hour. In 24×17, 17 denotes the number of features. These features do not include police department district id and temperature category.

TABLE 5. R-value of features with respect to the crime category.

Feature	R-Value
PdDistrict	-0.4262
Day of Week	0.3431
Day of Month	-0.0825
Month	0.1831
Year	-0.3669
$Cyclic_{xDay}$	-0.3652
$Cyclic_{yDay}$	0.5766
$Cyclic_{xWeek}$	0.5781
$Cyclic_{yWeek}$	-0.7431
temperature	-0.6560

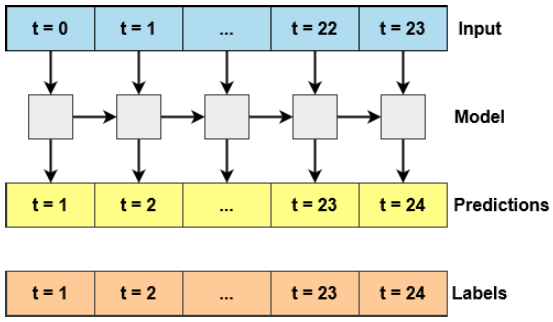


FIGURE 11. Data window generation for training, testing and validation.

E. TRAINING METHOD

Our main goal was to predict the number of each crime category using Spatio-temporal information. To do so, we trained the models according to Algorithm 1 using some evaluation metrics.

The generated windows are applied in the input of the models according to the requirements. Our model takes 24 sequential hours data and predicts the crimes in the next hour (25^{th} hour). After going through this algorithm, we have got the crime prediction of each category for ATTN-LSTM (Spatio-temporal), St-Bi-LSTM, and FLF models.

Here, Adam optimizer updates the gradients stochastically, and Early stopping avoids over-fitting the model. The models were evaluated based on the Mean Absolute Error (MAE), Mean Squared Error (MSE), Mean Squared Logarithmic Error (MSLE), Coefficient of Determination (R^2), and Symmetric Mean Absolute Percentage Error (SMAPE).

Algorithm 1 Training Strategy

Input : Training data (Classified Spatio-temporal features), α (patience)

Output : Spatio-temporal based prediction of each crime category for sub-models, Temporal based prediction for ATTN-LSTM model

Initialization: Initialize weights of Spatio-temporal based St-Bi-LSTM, ATTN-LSTM and Temporal based ATTN-LSTM model randomly; for initiating weights of Fusion model, use the weights of trained St-Bi-LSTM and ATTN-LSTM (only using temporal features) models

while *Maximum iteration is not reached* **do**

  Train St-Bi-LSTM and ATTN-LSTM Models: Generate output of given input data using swish activation; update the weights of the decoder stacked LSTM-network and the encoder dense-network using the gradient from the Adam optimizer and the loss function MAE given in (15);

Load Weights of FLF Model: Take the weights of St-Bi-LSTM and ATTN-LSTM (only using temporal features) models after trimming the last two layers and load them into FLF model;

while *Weights of St-Bi-LSTM and ATTN-LSTM models are not loaded in FLF model* **do**

  if *layer index is odd* **then**

    load this layer weights from St-Bi-LSTM model;

  else

    load this layer weights from ATTN-LSTM model;

Train the Fusion Model:

while *Maximum iteration is not reached* **do**

  Using Swish activation and Adam optimizer, update only the FLF model's last three layers (dense layers and the output layer) without changing the weights of other layers;

The Early stopping monitored the validation loss and stopped the training if the same loss occurred 10 times consecutively within 100 epochs. It also restored the model weights from the best value of validation loss. The MAE loss of Bayview for the training and validation dataset in the FLF module is given in Fig. 12. The amount of loss is decreasing with the increasing epochs. At the end of the training, the losses are near 0.02 for both data samples. Moreover, the MAE values for other areas are very low for training data. It indicates that our proposed model has good training accuracy on test and validation data.

F. EVALUATION CRITERIA

For performance evaluation, our architecture used Mean Absolute Error (MAE), Mean Squared Error (MSE), Mean Squared Logarithmic Error (MSLE), R^2 and Symmetric Mean Absolute Percentage Error (SMAPE) which are given in (15)−(19).

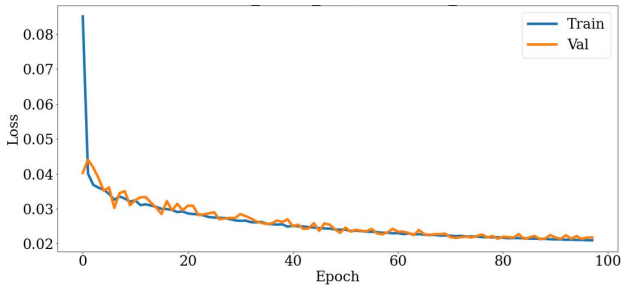


FIGURE 12. Training loss for a region of San Francisco using FLF module.

1) MEAN ABSOLUTE ERROR (MAE)

MAE is an estimator of the mean deviation of the observed value from actual values. For example, there are n data points in a sample, $\hat{y}$ and y represent the vector of prediction values and the vector of True values, respectively. Therefore, the equation for *MAE* is:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}(|\hat{y}_i - y_i|) \tag{15}$$

2) MEAN SQUARED ERROR (MSE)

It estimates the mean of the squared deviation of estimated values from the true values. Suppose there are n data points in a sample. We have generated the prediction vector for all the data of this sample. For this case, the *MSE* is computed as the equation given below:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{16}$$

3) MEAN SQUARED LOGARITHMIC ERROR (MSLE)

MSLE estimates the ratio between estimated values and the true values. In this case, the equation for *MSLE* is given below:

$$MSLE = \frac{1}{n}\sum_{i=1}^{n}(\log(y_i + 1) - \log(\hat{y}_i + 1))^2 \tag{17}$$

4) COEFFICIENT OF DETERMINATION (R^2)

R^2 is the ratio of the variation of the predicted variable. It indicates the closeness of actual and predicted values. Suppose the ratio of the sum of square regression (SSR) and the total sum of square (SST) tends towards zero, the fitness of a model increases. For n data points in a sample, $\bar{y}$ is the mean of all true values. In this case, the equation for R^2 is:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)}{\sum_{i=1}^{n}(y_i - \bar{y}_i)} \tag{18}$$

TABLE 6. Evaluation metrics of san francisco for proposed model.

Location	Sample of Data	MAE	MSLE	MSE	SMAPE(%)	R^2
BAYVIEW	Validation	0.0071	0.0018	0.0080	0.93	0.9616
	Test	0.0069	0.0018	0.0080	0.90	0.9616
CENTRAL	Validation	0.0081	0.0020	0.0104	0.97	0.9652
	Test	0.0083	0.0021	0.0156	0.95	0.9627
INGLESIDE	Validation	0.0060	0.0016	0.0071	0.79	0.9605
	Test	0.0054	0.0015	0.0064	0.71	0.9596
MISSION	Validation	0.0172	0.0050	0.0235	2.28	0.9276
	Test	0.0190	0.0056	0.0265	2.42	0.9312
NORTHERN	Validation	0.0106	0.0026	0.0147	1.24	0.956
	Test	0.0116	0.0029	0.0254	1.24	0.9485
PARK	Validation	0.0061	0.0016	0.0064	0.90	0.9470
	Test	0.0053	0.0014	0.0054	0.80	0.9483
RICHMOND	Validation	0.0042	0.0012	0.0045	0.62	0.9590
	Test	0.0044	0.0013	0.0051	0.61	0.9593
SOUTHERN	Validation	0.0116	0.0026	0.0210	1.15	0.9697
	Test	0.0108	0.0025	0.0169	1.10	0.9726
TARAVAL	Validation	0.0057	0.0015	0.0206	0.71	0.9410
	Test	0.0056	0.0016	0.0180	0.70	0.9501
TENDERLOIN	Validation	0.0053	0.0014	0.0063	0.73	0.9617
	Test	0.0043	0.0012	0.0047	0.64	0.9612

5) SYMMETRIC MEAN ABSOLUTE PERCENTAGE ERROR (SMAPE)

SMAPE is a percentage (or relative) error-based accuracy measure. It helps to analyze the sensitivity of seasonal time-series-based forecasting [52]. The model predicts better when the value of SMAPE is closer to zero. The equation of SMAPE is given below:

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{(y_i + \hat{y}_i)/2} * 100\% \tag{19}$$

G. FINAL OUTPUT GENERATION

The outputs of the ATTN-LSTM (where $nf = 19$), FLF module, and stream B are fused such that the DLF model had all the necessary information. This resulted in our model predicting each crime category's count better than the current state-of-arts.

In Algorithm 2, the way of merging the outputs of 3 different models in the DLF module is described. The model generates the final result by applying (7), (8) and calculates the final loss using (9). The value of n is 3 for these equations as there are 3 models to get the result of DLF. At first, this module checks if the inputs support the majority vote. Otherwise, it applies all-pairs shortest distance to get the final output. Then, the model calculated the loss of the final output using our weighted system. After doing all the steps according to this algorithm, the number of crimes per category for a specific hour and area is forecast.

Algorithm 2 Strategy of Final Prediction

Input : Prediction of the Spatio-temporal based st-Bi-LSTM, ATTN-LSTM, and FLF Module

Output : Spatio-temporal based prediction of each crime category using majority voting or all-pairs shortest distance on the DLF level

Prediction in DLF:

while *Each Input data* **do**

  Final Prediction: Get the final prediction using majority vote given in (7) or all-pairs shortest distance given in (8);

  if *Majority Vote of any prediction > threshold* **then**

    Final Result: Prediction of the most voted model;

  else

    Calculate Distance Matrix: Calculate all-pairs shortest distance for each model using (8) and store them in all-pairs loss matrix;

    Final Result: Prediction of the model which has the minimum average loss in the all-pairs loss matrix;

Calculate Loss for Proposed Model: Find the weight for each model, calculate the loss of each PdDistrict for predicted result using weighted average system given in (9), and calculate the loss for each city using (10);

VII. RESULT ANALYSIS

A. RESULT OF PROPOSED ARCHITECTURE

The idea of time-series forecasting with deep learning models was amalgamated to predict crime. All models were evaluated for each location of a city using MAE (15), MSE (16), MSLE (17), R^2 (18), and SMAPE (19). Finally, the evaluation metrics of each city for our proposed method using (9) and (10) is shown in Tables 6 and 7. Temperature is the derived feature in our model. Evaluation metrics for sub-models is listed in the appendix in Tables 9 and 10. We can observe the performance for the ATTN-LSTM model for temporal features, ATTN-LSTM model for all features,

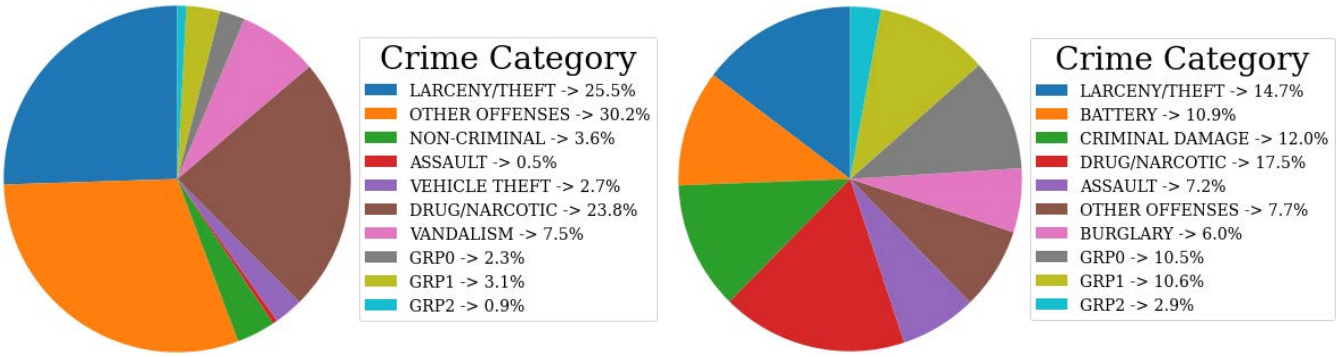
TABLE 7. Evaluation metrics of Chicago for proposed model.

Location of Chicago	Sample of Data	MAE	MSLE	MSE	SMAPE(%)	R^2
1.0	Validation	0.0252	0.0104	0.0271	1.20	0.847
	Test	0.0301	0.0124	0.0333	1.36	0.8637
2.0	Validation	0.0178	0.0069	0.0303	0.87	0.8722
	Test	0.0192	0.0077	0.0206	0.94	0.8470
3.0	Validation	0.0167	0.0057	0.0187	0.80	0.9123
	Test	0.0156	0.0053	0.0150	0.76	0.8959
4.0	Validation	0.0124	0.0027	0.0114	0.55	0.9456
	Test	0.0116	0.0026	0.0117	0.52	0.9663
5.0	Validation	0.0084	0.0020	0.0063	0.40	0.9542
	Test	0.0083	0.0020	0.0063	0.39	0.9536
6.0	Validation	0.0094	0.0018	0.0071	0.41	0.9670
	Test	0.0095	0.0019	0.0099	0.41	0.9544
7.0	Validation	0.0095	0.0017	0.0107	0.41	0.9422
	Test	0.0087	0.0016	0.0058	0.39	0.9645
8.0	Validation	0.0107	0.0019	0.0083	0.45	0.9679
	Test	0.0104	0.0019	0.0075	0.44	0.9666
9.0	Validation	0.0085	0.0016	0.0059	0.39	0.9628
	Test	0.0082	0.0015	0.0055	0.38	0.9611
10.0	Validation	0.0087	0.0015	0.0058	0.40	0.9613
	Test	0.0089	0.0016	0.006	0.40	0.9612
11.0	Validation	0.0125	0.0022	0.0256	0.52	0.9541
	Test	0.0117	0.0021	0.0093	0.50	0.9648
12.0	Validation	0.0094	0.0017	0.0064	0.45	0.9586
	Test	0.0100	0.0018	0.0071	0.47	0.9601
14.0	Validation	0.0080	0.0015	0.0051	0.41	0.9499
	Test	0.0084	0.0016	0.0056	0.42	0.9528
15.0	Validation	0.0097	0.0019	0.0074	0.50	0.9497
	Test	0.0090	0.0018	0.0061	0.48	0.9459
16.0	Validation	0.0093	0.0018	0.0063	0.50	0.9379
	Test	0.0093	0.0018	0.0064	0.50	0.9361
17.0	Validation	0.0084	0.0018	0.0057	0.48	0.9253
	Test	0.0087	0.0018	0.006	0.49	0.9264
18.0	Validation	0.0110	0.0023	0.0099	0.51	0.9346
	Test	0.0126	0.0025	0.0124	0.53	0.9424
19.0	Validation	0.0128	0.0027	0.0107	0.61	0.9248
	Test	0.0129	0.0028	0.0113	0.61	0.9257
20.0	Validation	0.0078	0.0018	0.0048	0.53	0.9468
	Test	0.0080	0.0019	0.0051	0.53	0.9474
22.0	Validation	0.0120	0.0032	0.0107	0.63	0.9505
	Test	0.0119	0.0032	0.0106	0.62	0.9501
24.0	Validation	0.0113	0.0030	0.0094	0.62	0.9497
	Test	0.0118	0.0032	0.0103	0.63	0.9482
25.0	Validation	0.0180	0.0050	0.0202	0.78	0.9533
	Test	0.0175	0.0049	0.0195	0.77	0.9505

St-Bi-LSTM model, Feature level Fusion model, and proposed model from these tables.

In the Spatio-temporal based analysis on the San Francisco dataset, ATTN-LSTM performed well in Bayview based on MAE. At the same time, St-Bi-LSTM worked well in Mission and Northern. The MAE loss for these two models in these three places is 0.02. However, the R^2 values for St-Bi-LSTM for all these three places are highest. For Bayview, Mission, and Northern, the values of R^2 for this model are 0.967, 0.97, and 0.971, respectively. For the rest of the locations, the Fusion model performed better among the sub-models shown in Table 9. The MAE loss is from 0.0101 to 0.0299 and the R^2 value is from 0.9643 to 0.9765 for those places. However, the values of SMAPE are the lowest for all locations in the fusion model. The ATTN-LSTM took the most relevant instances while St-Bi-LSTM chunked the irrelevant ones off while training. Hence, these two models contributed to the improved outcome of the Fusion model of the feature level. In other words, both of the models were important in the case of prediction using a large dataset.

For this dataset, the temporal-based ATTN-LSTM model had the highest R^2, MAE and SMAPE value concerning Spatio-temporal based models. The values of R^2, MAE and SMAPE are 0.988, 0.12, and around 9%, respectively. Here,



(a) Predicted crime for Bayview of San Francisco

(b) Predicted crime for Area-2.0 of Chicago

FIGURE 13. Some examples of crime prediction of a specific hour and specific location for the proposed model.

TABLE 8. Comparative analysis with some of the current state-of-the-art for crime datasets dating back more than a decade.

Models Proposed By		Sample of Data	Evaluation metrics of San Francisco					Evaluation metrics of Chicago				
			MAE	MSLE	MSE	SMAPE	R^2	MAE	MSLE	MSE	SMAPE	R^2
Non-DL	SARIMAX(1,1,1,2)	Validation	1.2531	0.4750	2.7581	100.00	-0.0061	1.0906	0.3742	2.1081	100.00	-0.0580
		Test	1.2408	0.4483	2.8146	100.00	-0.0063	1.2309	0.3164	2.6379	100.00	-0.0196
	FBProphet	Validation	1.2856	0.3631	3.1071	103.24	0.1245	0.9590	0.2783	1.6210	105.76	0.0867
		Test	1.2816	0.3727	3.0382	104.81	0.1085	0.9959	0.2874	1.7937	109.12	0.0245
DL based	Feng et al. [1]	Validation	0.4477	0.0910	0.3214	58.35	0.2549	1.8344	0.2544	9.7052	62.27	0.1997
		Test	0.4442	0.0901	0.3202	58.35	0.2520	1.9376	0.2932	9.6717	64.61	0.1945
	Rayhan et al. [9]	Validation	0.2382	0.0457	0.1870	20.25	0.5624	0.5667	0.1926	0.9250	47.86	0.3528
		Test	0.2320	0.0443	0.1845	20.20	0.5606	0.5678	0.1873	0.9425	47.72	0.3516
	Wang et al. [38]	Validation	0.0288	0.0068	0.0306	3.34	0.8538	0.0816	0.0462	0.1301	4.83	0.7516
		Test	0.0284	0.0067	0.0297	3.31	0.8502	0.0824	0.0461	0.1294	4.80	0.7509
	Our Proposed Model	**Validation**	**0.0082**	**0.0021**	**0.0123**	**1.03**	**0.9549**	**0.0201**	**0.008**	**0.0267**	**0.57**	**0.9416**
		Test	**0.0082**	**0.0022**	**0.0132**	**1.01**	**0.9550**	**0.0202**	**0.008**	**0.0278**	**0.57**	**0.9376**

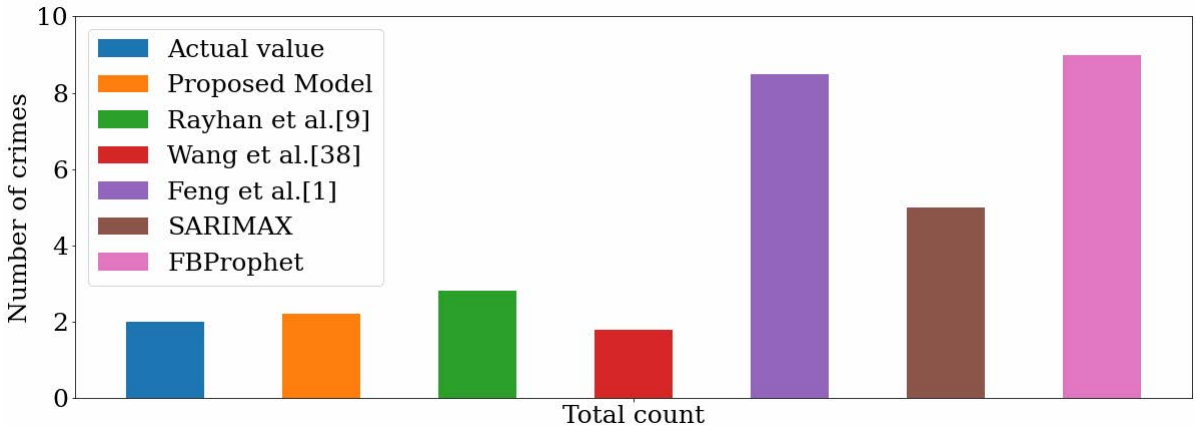


FIGURE 14. The actual number of crime occurrence and prediction of the number of crime occurrence of an specific hour for Area-6.0 of Chicago (using some current state-of-the-arts and our proposed model).

we took the whole city as one location. However, the crime patterns are different for areas under a city.

The evaluation results of our proposed model for each location of San Francisco are given in Table 6. After observing the outcomes of both tables, we concluded that the loss in the proposed model for this dataset is relatively less than the rest of the sub-models. The MAE loss of test and validation dataset in crime prediction for each area is less than 0.02 here. In the case of MSLE and MSE, the values of the proposed model for each area are less than 0.003 and 0.027, respectively. These values are way lower than the sub-models. The SMAPE value for each location is also less than the sub-models. This value is between 0.61% to 2.42%. Our model fused the categorical, temporal, and spatial information by

TABLE 9. Evaluation metrics of San Francisco for sub-models.

Location	Derived Feature	Model	Dataset	MAE	MSLE	MSE	SMAPE(%)	R^2
San Francisco		ATTN-LSTM	Validation	0.1240	0.0199	0.3888	9.21	0.9874
			Test	0.1185	0.0194	0.3743	8.60	0.9883
BAYVIEW	Temperature	ATTN-LSTM	Validation	**0.0207**	0.0061	0.0272	2.89	0.9569
			Test	**0.0199**	0.0061	0.0271	2.77	0.9567
		St-Bi-LSTM	Validation	0.0244	**0.0048**	**0.021**	3.14	**0.9665**
			Test	0.0224	**0.0048**	**0.0207**	2.92	**0.9669**
		Fusion	Validation	0.0213	**0.0048**	**0.021**	**2.56**	0.9663
			Test	0.0212	**0.0048**	0.0212	**2.56**	0.9663
CENTRAL	Temperature	ATTN-LSTM	Validation	0.0242	0.0064	0.0336	3.07	0.9625
			Test	**0.0245**	0.0067	0.0584	2.93	0.958
		St-Bi-LSTM	Validation	0.0283	**0.0056**	**0.028**	3.69	**0.9687**
			Test	0.0282	**0.0059**	**0.0335**	3.55	**0.9684**
		Fusion	Validation	**0.0232**	0.0057	0.0289	**2.49**	0.9676
			Test	0.0247	0.006	0.0352	**2.55**	0.9673
INGLESIDE	Temperature	ATTN-LSTM	Validation	0.0184	0.0055	0.0239	2.57	0.956
			Test	0.0162	0.0051	0.0217	2.28	0.9544
		St-Bi-LSTM	Validation	0.0229	0.0043	0.0191	2.99	0.9647
			Test	0.0198	0.004	0.0172	2.64	0.964
		Fusion	Validation	**0.0166**	**0.0042**	**0.0189**	**1.98**	**0.9651**
			Test	**0.015**	**0.0039**	**0.0167**	**1.83**	**0.9651**
MISSION	Temperature	ATTN-LSTM	Validation	0.0804	0.0242	0.1118	11.22	0.8863
			Test	0.0884	0.027	0.1246	11.89	0.8928
		St-Bi-LSTM	Validation	**0.0213**	**0.0059**	**0.028**	2.5	**0.9703**
			Test	**0.0234**	**0.0065**	**0.032**	2.68	**0.9721**
		Fusion	Validation	0.0229	0.006	0.0299	**2.44**	0.9685
			Test	0.0258	0.0069	0.0354	**2.66**	0.9691
NORTHERN	Temperature	ATTN-LSTM	Validation	0.0424	0.0098	0.0579	5.12	0.9425
			Test	0.0465	0.0108	0.1157	5.07	0.9276
		St-Bi-LSTM	Validation	**0.0215**	**0.0057**	**0.0292**	2.5	**0.9707**
			Test	**0.0229**	**0.0061**	**0.0352**	2.57	**0.971**
		Fusion	Validation	**0.0215**	0.006	0.0309	**2.25**	0.9691
			Test	0.0235	0.0066	0.0373	**2.35**	0.9689
PARK	Temperature	ATTN-LSTM	Validation	0.0251	0.0063	0.0255	3.87	0.9294
			Test	0.0217	0.0057	0.0217	3.39	0.9316
		St-Bi-LSTM	Validation	0.0127	**0.0032**	0.0128	1.7	**0.9647**
			Test	0.0112	**0.003**	0.011	1.58	**0.9652**
		Fusion	Validation	**0.0111**	**0.0032**	**0.0126**	**1.46**	**0.9647**
			Test	**0.0101**	**0.003**	**0.0109**	**1.38**	0.9649
RICHMOND	Temperature	ATTN-LSTM	Validation	0.0149	0.0041	0.0157	2.29	0.953
			Test	0.0152	0.0044	0.0177	2.23	0.9533
		St-Bi-LSTM	Validation	0.0118	**0.003**	0.0116	1.65	**0.9651**
			Test	0.0125	**0.0032**	0.0131	1.68	**0.9652**
		Fusion	Validation	**0.0101**	**0.003**	**0.0115**	**1.35**	0.965
			Test	**0.0109**	**0.0032**	**0.013**	**1.4**	**0.9652**
SOUTHERN	Temperature	ATTN-LSTM	Validation	0.0391	0.0086	0.0801	4.04	0.9629
			Test	0.0352	0.0082	0.0591	3.77	0.9683
		St-Bi-LSTM	Validation	0.0332	**0.0068**	**0.0436**	3.32	**0.9776**
			Test	0.0316	**0.0066**	**0.0401**	3.22	**0.978**
		Fusion	Validation	**0.0299**	0.0069	0.0466	**2.74**	0.9761
			Test	**0.0288**	0.0067	0.0428	**2.69**	0.9765
TARAVAL	Temperature	ATTN-LSTM	Validation	0.0204	0.0055	0.1072	2.61	0.9171
			Test	0.0195	0.0058	0.0909	2.46	0.9347
		St-Bi-LSTM	Validation	0.0169	0.0038	0.0169	2.1	0.9641
			Test	0.0164	0.004	0.0179	2.03	0.9647
		Fusion	Validation	**0.0127**	**0.0037**	**0.0161**	**1.58**	**0.9652**
			Test	**0.0133**	**0.0039**	**0.0173**	**1.63**	**0.9657**

TABLE 9. *(Continued.)* **Evaluation metrics of San Francisco for sub-models.**

Location	Derived Feature	Model	Dataset	MAE	MSLE	MSE	SMAPE(%)	R^2
TENDERLOIN	Temperature	ATTN-LSTM	Validation	0.0172	0.0047	0.0207	2.61	0.9583
			Test	0.014	0.0039	0.0153	2.27	0.9582
		St-Bi-LSTM	Validation	0.0192	0.004	0.0176	2.35	0.9644
			Test	0.0154	0.0034	0.0133	2.01	0.9635
		Fusion	Validation	**0.0132**	**0.0039**	**0.017**	**1.62**	**0.9653**
			Test	**0.011**	**0.0033**	**0.0128**	**1.44**	**0.9643**

using 3 different sub-models in the DLF. This module took the best outcome from those sub-models using majority vote (7) and all pair shortest distance (8). We calculated the weighted average loss (9) of this module based on the inputs. So, the loss is less than the rest of the models. The value of R^2 for each location of this city is between 92% to 97%. So, our model worked as intended for the San Francisco dataset.

The loss of proposed model and sub-models for the Chicago dataset are given in Tables 7, 10. In the case of this dataset, ATTN-LSTM performed better for some places. St-Bi-LSTM using the transfer learning technique has fared better for the remaining ones among the sub-models. After comparing the values of MAE, MSLE, MSE, SMAPE and R^2, it is clear that our proposed model worked better than most of the sub-models for this dataset. The losses and the R^2 value are better here for the same reason as the San Francisco dataset. For the proposed model, the value of MAE is less than 0.035, MSLE is less than 0.015, MSE is less than 0.55, SMAPE is between 0.4 to 1.36 and R^2 is between 84% to 98% for each location.

The average loss for each city is calculated by combining the losses of each region using (10). These values are listed in Table 8. For the San Francisco dataset, the cumulative average MAE loss is 0.0082, MSLE loss is 0.002, MSE losses are 0.0132 (test data), 0.0123 (validation data), SMAPE is 1.03 and R^2 is 0.955. The cumulative average MAE loss is 0.02, MSLE loss is 0.008, MSE loss is 0.027, SMAPE is 0.57 and R^2 is 0.94 for the Chicago dataset. From these values, we can state that our model had a satisfactory amount of losses and R^2. Thus, introducing the DLF module into the model was a good decision.

A transfer learning technique is implemented to train some features in the Chicago dataset. In this case, the source model is the Attn-model of the San-Francisco dataset. By applying this technique, the training time is decreased by 30 minutes, considering the other sub-models.

If any of the sub-models are not added to our model, the model would not co-relate the different types of features, avoid unnecessary instances and emphasis on related instances, or reduce training time through learning convergence. Furthermore, our goal of making a generalized model would not be fulfilled.

Some predictions of our model are given in Fig. 13. The predictions for an area are plotted using a pie chart here. Fig. 13a is the prediction of crime for Bayview of San Francisco. Fig. 13b is the visualization of the crime prediction for Area-2.0 of Chicago. In these figures, one can observe the predicted percentage of each crime category for an hour for a specific area of both cities. Here, the "Other offenses" percentage is highest for Bayview of San Francisco, and "Drug/Narcotic" is highest for Area-2.0 of Chicago. From these percentages, the law enforcers can know the crime rate for each category for an hour. They will be able to take necessary precautions to reduce the crime occurrence. Our model can also predict the number of crimes per category for these areas.

B. COMPARATIVE STUDY

In Table 8, the evaluation metrics for the San Francisco dataset and Chicago dataset for some established models and our proposed model can be observed. Some renowned DL and non-DL-based methods are implemented to illustrate this comparative study. The models are evaluated on the same train, test, and validation sets. In case of Dl-based method, the model of Feng *et al.* has the highest loss, Rayhan *et al.*'s model has the second-highest, and our proposed model has the last for both datasets. Our proposed model has an MAE loss of 0.008 and 0.02 for the two datasets, but other models have error near 0.44 and 1.83 [1], 0.24 and 0.57 [9], 0.03 and 0.08 [38]. For both datasets, the MAE values for SARIMAX are close to 1.2 and for FBProphet are close to 1.3 (San Francisco) and 1.0 (Chicago). From these results, we can perceive that our proposed model has the smallest loss, and it performs better among these 6 models. Moreover, our proposed model has the highest R^2 value among the 6 models. The values are approximately 0.95 and 0.94 for both datasets in our model. For other models, the values are 0.25 and 0.19 [1], near 0.56 and 0.35 [9], 0.85 and 0.75 [38], 0.1 and 0.09 (validation), 0.03 (test) (FBProphet), -0.006 and -0.05(validation), -0.02 (test) (SARIMAX) for the San Francisco and Chicago dataset, respectively. It is known that the closer the value of R^2 to 1, the better the model predicts the values. Since the R^2 of our proposed model is closer to 1, we can claim that the proposed model predicts better than the current state-of-art. The SMAPE values for our proposed model are the lowest of all other models. It proves that the model is predicting better than the rest of the models. Our proposed model has SMAPE values of 1 and 1.5 for both dataset.

TABLE 10. Evaluation metrics of Chicago for sub-models.

Location	Derived Feature	Model	Dataset	MAE	MSLE	MSE	SMAPE(%)	R^2
Chicago		ATTN-LSTM	Validation	1.3049	0.2063	3.8324	9.57	0.9597
			Test	1.3166	0.2174	3.7001	9.01	0.9598
1.0	Temperature	ATTN-LSTM	Validation	**0.0252**	**0.0047**	**0.0197**	**4.45**	**0.9617**
			Test	**0.029**	**0.0053**	**0.0309**	**4.63**	**0.9652**
		St-Bi-LSTM	Validation	0.0777	0.0339	0.0886	12.62	0.8235
			Test	0.0923	0.0399	0.1154	14.37	0.8435
		Fusion	Validation	0.1102	0.0359	0.0852	14.31	0.9207
			Test	0.138	0.0466	0.0437	16.69	0.9234
2.0	Temperature	ATTN-LSTM	Validation	**0.0151**	**0.0047**	**0.0166**	**2.46**	**0.9558**
			Test	**0.0151**	**0.0049**	**0.017**	**2.41**	**0.9578**
		St-Bi-LSTM	Validation	0.0533	0.0217	0.0551	8.98	0.8557
			Test	0.058	0.024	0.0603	9.73	0.8507
		Fusion	Validation	0.091	0.0302	0.4527	13.19	0.9207
			Test	0.0982	0.0332	0.1196	14.19	0.7069
3.0	Temperature	ATTN-LSTM	Validation	**0.0258**	**0.0052**	**0.0187**	**5.05**	**0.9604**
			Test	**0.0232**	**0.005**	**0.0173**	**4.5**	**0.9599**
		St-Bi-LSTM	Validation	0.0469	0.0166	0.0447	7.67	0.9052
			Test	0.0436	0.0151	0.0397	7.16	0.9076
		Fusion	Validation	0.0993	0.0329	0.1848	13.97	0.9207
			Test	0.0964	0.0325	0.1135	13.81	0.7379
4.0	Temperature	ATTN-LSTM	Validation	0.0323	0.006	0.0236	5.72	0.962
			Test	0.0274	0.0056	0.0209	5.1	0.9616
		St-Bi-LSTM	Validation	**0.0286**	**0.0048**	**0.0181**	**4.28**	**0.9708**
			Test	**0.0269**	**0.0046**	**0.0165**	**4.07**	**0.9695**
		Fusion	Validation	0.1106	0.0368	0.1742	14.93	0.7269
			Test	0.1052	0.0356	0.1986	14.48	0.9457
5.0	Temperature	ATTN-LSTM	Validation	0.0352	0.0051	0.0181	6.48	0.9557
			Test	0.0329	0.005	0.0179	5.87	0.9563
		St-Bi-LSTM	Validation	**0.0183**	**0.0037**	**0.0125**	**2.92**	**0.9697**
			Test	**0.0179**	**0.0037**	**0.0123**	**2.88**	**0.9694**
		Fusion	Validation	0.0717	0.0245	0.0718	10.15	0.8285
			Test	0.0737	0.0255	0.0727	10.52	0.8242
6.0	Temperature	ATTN-LSTM	Validation	0.0536	0.0071	0.0297	7.89	0.9543
			Test	0.0539	0.0071	0.0294	7.74	0.9545
		St-Bi-LSTM	Validation	**0.0225**	**0.0047**	**0.0178**	**3.45**	**0.9723**
			Test	**0.0226**	**0.0048**	**0.0177**	**3.46**	**0.9724**
		Fusion	Validation	0.0483	0.0096	0.0408	5.63	0.9372
			Test	0.0502	0.0101	0.1258	5.91	0.8103
7.0	Temperature	ATTN-LSTM	Validation	0.0674	0.0081	0.0331	9.9	0.9451
			Test	0.0635	0.0075	0.0295	9.65	0.9404
		St-Bi-LSTM	Validation	**0.0221**	**0.0045**	**0.017**	**3.35**	**0.9719**
			Test	**0.0204**	**0.0042**	**0.015**	**3.15**	**0.9694**
		Fusion	Validation	0.0399	0.0072	0.1506	4.5	0.7021
			Test	0.035	0.0065	0.0257	4.16	0.9492
8.0	Temperature	ATTN-LSTM	Validation	0.0938	0.0106	0.0547	12.76	0.9334
			Test	0.0906	0.0102	0.0433	12.71	0.9351
		St-Bi-LSTM	Validation	**0.0233**	**0.0049**	**0.0199**	**3.41**	**0.9734**
			Test	**0.0226**	**0.0048**	**0.0188**	**3.31**	**0.9719**
		Fusion	Validation	0.0418	0.0071	0.0338	4.47	0.9587
			Test	0.0409	0.007	0.0306	4.44	0.9553
9.0	Temperature	ATTN-LSTM	Validation	0.0915	0.0103	0.0439	14.13	0.9098
			Test	0.0894	0.01	0.0384	14.12	0.9068
		St-Bi-LSTM	Validation	**0.0177**	**0.004**	**0.0139**	2.76	**0.9701**
			Test	**0.0168**	**0.0038**	**0.0129**	2.64	**0.9687**
		Fusion	Validation	0.0219	0.0045	0.0213	**2.47**	0.9574
			Test	0.0209	0.0044	0.0225	**2.41**	0.9551
10.0	Temperature	ATTN-LSTM	Validation	0.101	0.0118	0.0488	16.73	0.8919
			Test	0.1054	0.0123	0.0519	17.2	0.8881
		St-Bi-LSTM	Validation	**0.0176**	**0.0038**	**0.0132**	2.61	**0.9698**
			Test	**0.0176**	**0.0039**	**0.0134**	2.61	**0.9699**
		Fusion	Validation	0.0185	0.0041	0.0191	**2.12**	0.963
			Test	0.0191	0.0041	0.0197	**2.18**	0.9642
11.0	Temperature	ATTN-LSTM	Validation	0.1436	0.0185	0.1046	22.8	0.9019
			Test	0.141	0.0179	0.0854	22.28	0.8924
		St-Bi-LSTM	Validation	**0.024**	**0.0049**	**0.0231**	**3.15**	**0.9761**
			Test	**0.0222**	**0.005**	**0.0204**	**3.07**	**0.9737**
		Fusion	Validation	0.04	0.0065	0.4779	3.6	0.8302
			Test	0.0325	0.0056	0.0303	3.41	0.9653

TABLE 10. *(Continued.)* **Evaluation metrics of Chicago for sub-models.**

12.0	Temperature	ATTN-LSTM	Validation	0.1307	0.0157	0.0664	22.18	0.8541
			Test	0.1365	0.0162	0.0713	22.61	0.8637
		St-Bi-LSTM	Validation	**0.0167**	**0.0038**	**0.0134**	2.65	**0.9708**
			Test	**0.018**	**0.004**	**0.0151**	2.78	**0.9716**
		Fusion	Validation	0.0175	0.004	0.0174	**2.01**	0.9657
			Test	0.0199	0.0043	0.0205	**2.16**	0.9653
14.0	Temperature	ATTN-LSTM	Validation	0.1263	0.0164	0.0618	23.99	0.7981
			Test	0.1312	0.0167	0.065	24.32	0.8148
		St-Bi-LSTM	Validation	0.0126	**0.0031**	**0.01**	1.98	**0.9672**
			Test	**0.0132**	**0.0033**	**0.0111**	2.02	**0.9687**
		Fusion	Validation	**0.0121**	0.0032	0.0121	**1.55**	0.9637
			Test	0.0136	0.0034	0.0144	**1.67**	0.9635
15.0	Temperature	ATTN-LSTM	Validation	0.1569	0.025	0.094	30.1	0.7824
			Test	0.1504	0.0234	0.0815	29.48	0.7552
		St-Bi-LSTM	Validation	**0.0149**	**0.0036**	**0.0125**	2.22	**0.971**
			Test	0.0134	**0.0034**	**0.0107**	2.09	**0.968**
		Fusion	Validation	0.0155	0.0038	0.028	**1.89**	0.9463
			Test	**0.0133**	**0.0034**	0.0156	**1.73**	0.9598
16.0	Temperature	ATTN-LSTM	Validation	0.1626	0.0273	0.0975	31.86	0.677
			Test	0.1605	0.0271	0.0967	31.65	0.6677
		St-Bi-LSTM	Validation	0.0133	**0.0031**	**0.0098**	2.09	**0.9677**
			Test	0.0134	**0.003**	**0.0096**	2.08	**0.9672**
		Fusion	Validation	**0.0111**	0.0032	0.0147	**1.47**	0.9602
			Test	**0.011**	0.0031	0.0174	**1.44**	0.9563
17.0	Temperature	ATTN-LSTM	Validation	0.1638	0.0305	0.0988	33.93	0.5606
			Test	0.1667	0.0305	0.1043	33.95	0.5699
		St-Bi-LSTM	Validation	0.0101	**0.0026**	**0.0076**	1.63	**0.9664**
			Test	0.0105	**0.0027**	**0.0082**	1.66	**0.9664**
		Fusion	Validation	**0.0086**	**0.0026**	0.0105	**1.24**	0.961
			Test	**0.0089**	**0.0027**	0.0101	**1.28**	0.9633
18.0	Temperature	ATTN-LSTM	Validation	0.2124	0.0408	0.1814	35.76	0.5907
			Test	0.2363	0.0444	0.2229	36.22	0.6495
		St-Bi-LSTM	Validation	**0.013**	**0.0031**	**0.012**	1.76	**0.9733**
			Test	**0.0156**	**0.0035**	**0.016**	1.94	**0.9753**
		Fusion	Validation	**0.013**	0.0033	0.0203	**1.55**	0.9685
			Test	0.0163	0.0037	0.02	**1.74**	0.9717
19.0	Temperature	ATTN-LSTM	Validation	0.2365	0.0509	0.2057	40.39	0.5127
			Test	0.2402	0.0512	0.2158	40.15	0.5182
		St-Bi-LSTM	Validation	0.0168	**0.0034**	**0.0123**	2.39	**0.9714**
			Test	0.0168	**0.0035**	**0.013**	2.37	**0.9716**
		Fusion	Validation	**0.0132**	0.0037	0.0176	**1.64**	0.9647
			Test	**0.0137**	0.0038	0.0184	**1.65**	0.9661
20.0	Temperature	ATTN-LSTM	Validation	0.1756	0.0398	0.1058	41.36	0.8176
			Test	0.1792	0.0409	0.1111	41.29	0.8211
		St-Bi-LSTM	Validation	0.0067	**0.0017**	**0.0043**	1.36	**0.9615**
			Test	0.0069	**0.0018**	**0.0045**	1.37	**0.9623**
		Fusion	Validation	**0.0053**	**0.0017**	0.0047	**0.93**	0.9587
			Test	**0.0055**	**0.0018**	0.0054	**0.95**	0.955
22.0	Temperature	ATTN-LSTM	Validation	0.2652	0.0701	0.2367	48.51	0.7995
			Test	0.2642	0.0694	0.2351	48.37	0.8022
		St-Bi-LSTM	Validation	0.0106	**0.0029**	**0.0089**	1.6	**0.9678**
			Test	0.0104	0.0029	**0.0088**	1.57	**0.9669**
		Fusion	Validation	**0.0089**	**0.0029**	0.0123	**1.28**	0.9634
			Test	**0.0089**	**0.0028**	0.0114	**1.27**	0.9632
24.0	Temperature	ATTN-LSTM	Validation	0.2562	0.0689	0.2162	48.78	0.7995
			Test	0.2663	0.0712	0.2347	48.76	0.7823
		St-Bi-LSTM	Validation	0.0092	**0.0025**	**0.0072**	1.55	**0.9665**
			Test	0.0098	**0.0027**	**0.0081**	1.59	**0.9668**
		Fusion	Validation	**0.0075**	**0.0025**	0.0075	**1.17**	0.9653
			Test	**0.0081**	**0.0027**	0.0089	**1.22**	0.9653
25.0	Temperature	ATTN-LSTM	Validation	0.3853	0.1089	0.4599	57.43	0.7843
			Test	0.3797	0.1065	0.4441	56.76	0.7669
		St-Bi-LSTM	Validation	0.0175	**0.0045**	**0.0163**	2.35	**0.9721**
			Test	0.0165	**0.0043**	**0.0154**	2.25	**0.971**
		Fusion	Validation	**0.0148**	**0.0045**	0.0167	**1.85**	0.9716
			Test	**0.0142**	0.0044	0.0163	**1.79**	0.9702

The predicted number of crimes for a specific hour for all the 6 models can be distinguished from Fig. 14. The blue bar denotes the true/actual value. We took Area-6.0 of Chicago for this visualization. This visualization shows that our model predicted the number of crimes closer to the true value. The actual value is 2. Our model predicted a little higher than the actual value. In contrast, the model of Wang *et al.* [38] predicted a little lower than the actual value. The attention-based model of Rayhan *et al.* [9] predicted close to 3. The SARIMAX model predicted 5 crimes for the specific hour. The models of Feng *et al.* [1] and FBProphet predicted the highest value among these 6 models. The predicted total number of crimes is approximately 9 for these models. Hence, this model does not perform well. Based on this prediction, we can conclude that our model outperformed the other 5 models.

The mentioned state-of-arts have limitations regarding the length of data [1], [9], [38]. Our model can manage data from over a decade without losing any plausibility. We employed the ATTN-LSTM model and the transfer learning technique in the St-Bi-LSTM model to deal with this problem.

A few models [1] did not consider Spatio features of a city. We utilized the knowledge of a city's districts or regions and categorical-temporal information to forecast crime. So, our model has no limitations regarding Spatio features.

All four models have FLF in the architecture. However, our model has two Fusion module levels- FLF and DLF. The FLF module merges the Spatio-temporal features along with the categorical information. The model learns more about the different outcomes from Spatio-temporal based sub-models in the DLF module. Hence, the model can predict more accurately the number of crimes for each category and each location with the help of this learning. Finally, we can state that our proposed model is an effective method for forecasting data.

VIII. CONCLUSION AND FUTURE WORK

In this work, the fusion technique is applied to predict crime on an hourly timescale for two cities in the USA. Introducing the DLF module into our architecture helped to predict the best result. Moreover, the use of the Transfer learning technique reduced the training time to a certain amount. Our model can predict crime from Spatio-temporal based Categorical data and has overcome all the limitations of the current state-of-the-art.

Although our model performs well, there are some disadvantages, such as the training time of the whole system being more than moderate. Due to the small amount of data in some categories, we need to re-categorize them into 3 groups.

In the future, to overcome these problems, we plan to develop a model that requires less time to train and can also work with a small amount of data in a category.

ACKNOWLEDGMENT

The authors would like to thank all the people who have given their suggestions to achieve a great result. They also want to thank the reviewers for their precious opinions on this work.

CONFLICTS OF INTEREST

The authors indicate that there are no conflicts of interest with relation to the publication.

APPENDIX

RESULTS OF SUB-MODELS

See Tables 9 and 10.

REFERENCES

[1] M. Feng, J. Zheng, J. Ren, A. Hussain, X. Li, Y. Xi, and Q. Liu, "Big data analytics and mining for effective visualization and trends forecasting of crime data," *IEEE Access*, vol. 7, pp. 106111–106123, 2019.

[2] M. Feng, J. Zheng, Y. Han, J. Ren, and Q. Liu, "Big data analytics and mining for crime data analysis, visualization and prediction," in *Proc. Int. Conf. Brain Inspired Cognit. Syst.* Cham, Switzerland: Springer, 2018, pp. 605–614.

[3] E. R. Groff and N. G. L. Vigne, "Forecasting the future of predictive crime mapping," *Crime Prevention Stud.*, vol. 13, pp. 29–58, Jan. 2002.

[4] H. J. Eysenck, "Crime and personality," *Medico-Legal J.*, vol. 47, no. 1, pp. 18–32, 1979.

[5] W. Safat, S. Asghar, and S. A. Gillani, "Empirical analysis for crime prediction and forecasting using machine learning and deep learning techniques," *IEEE Access*, vol. 9, pp. 70080–70094, 2021.

[6] N. Jin, Y. Zeng, K. Yan, and Z. Ji, "Multivariate air quality forecasting with nested long short term memory neural network," *IEEE Trans. Ind. Informat.*, vol. 17, no. 12, pp. 8514–8522, Dec. 2021.

[7] Y.-L. Hu and L. Chen, "A nonlinear hybrid wind speed forecasting model using LSTM network, hysteretic ELM and differential evolution algorithm," *Energy Convers. Manage.*, vol. 173, pp. 123–142, Oct. 2018.

[8] H. Abbasimehr, M. Shabani, and M. Yousefi, "An optimized model using LSTM network for demand forecasting," *Comput. Ind. Eng.*, vol. 143, May 2020, Art. no. 106435.

[9] Y. Rayhan and T. Hashem, "AIST: An interpretable attention-based deep learning model for crime prediction," 2020, *arXiv:2012.08713*.

[10] M. C. Bishop, "Multi layer perceptron," in *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995, pp. 116–163.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[13] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," 2016, *arXiv:1601.06733*.

[14] Q.-L. Ma, Q.-L. Zheng, H. Peng, T.-W. Zhong, and L.-Q. Xu, "Chaotic time series prediction based on evolving recurrent neural networks," in *Proc. Int. Conf. Mach. Learn. Cybern.*, Aug. 2007, pp. 3496–3500.

[15] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[16] A. B. Said, A. Erradi, H. A. Aly, and A. Mohamed, "Predicting COVID-19 cases using bidirectional LSTM on multivariate time series," *Environ. Sci. Pollut. Res.*, vol. 28, no. 40, pp. 56043–56052, Oct. 2021.

[17] J. Kim and N. Moon, "BiLSTM model based on multivariate time series data in multiple field for forecasting trading area," *J. Ambient Intell. Hum. Comput.*, vol. 15, pp. 1–10, Jul. 2019, doi: 10.1007/s12652-019-01398-9.

[18] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction," 2018, *arXiv:1801.02143*.

[19] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values," *Transp. Res. C, Emerg. Technol.*, vol. 118, Sep. 2020, Art. no. 102674.

[20] K. Hetterscheid, "Detecting agitated speech: A neural network approach," M.S. thesis, Univ. Twente, Enschede, The Netherlands, 2020.

[21] Y. Zhao, L. Ye, Z. Li, X. Song, Y. Lang, and J. Su, "A novel bidirectional mechanism based on time series model for wind power forecasting," *Appl. Energy*, vol. 177, pp. 793–803, Sep. 2016.

[22] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.

[23] Y. Kim, C. Denton, L. Hoang, and A. M. Rush, "Structured attention networks," 2017, *arXiv:1702.00887*.
[24] H. Abbasimehr and R. Paki, "Improving time series forecasting using LSTM and attention models," *J. Ambient Intell. Hum. Comput.*, vol. 13, pp. 1–19, Jan. 2021.
[25] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A decomposable attention model for natural language inference," 2016, *arXiv:1606.01933*.
[26] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015, *arXiv:1508.04025*.
[27] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*.
[28] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.
[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[30] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: The difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Networks*, S. C. Kremer and J. F. Kolen, eds., IEEE Press, 2001.
[31] A. Pandey and D. Wang, "Dense CNN with self-attention for time-domain speech enhancement," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 29, pp. 1270–1279, Mar. 2021.
[32] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," 2014, *arXiv:1412.7755*.
[33] S. Merity, "Single headed attention RNN: Stop thinking with your head," 2019, *arXiv:1911.11423*.
[34] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," 2019, *arXiv:1906.05909*.
[35] B. Chen, J. Li, X. Guo, and G. Lu, "DualCheXNet: Dual asymmetric feature learning for thoracic disease classification in chest X-rays," *Biomed. Signal Process. Control*, vol. 53, Aug. 2019, Art. no. 101554.
[36] J. Li, B. Zhang, G. Lu, and D. Zhang, "Dual asymmetric deep hashing learning," *IEEE Access*, vol. 7, pp. 113372–113384, 2019.
[37] B. Wang, P. Yin, A. L. Bertozzi, P. J. Brantingham, S. J. Osher, and J. Xin, "Deep learning for real-time crime forecasting and its ternarization," *Chin. Ann. Math., B*, vol. 40, no. 6, pp. 949–966, Nov. 2019.
[38] B. Wang, D. Zhang, D. Zhang, P. J. Brantingham, and A. L. Bertozzi, "Deep learning for real time crime forecasting," 2017, *arXiv:1707.03340*.
[39] C.-H. Yu, M. W. Ward, M. Morabito, and W. Ding, "Crime forecasting using data mining techniques," in *Proc. IEEE 11th Int. Conf. Data Mining Workshops*, Dec. 2011, pp. 779–786.
[40] J. Agarwal, R. Nagpal, and R. Sehgal, "Crime analysis using K-Means clustering," *Int. J. Comput. Appl.*, vol. 83, no. 4, pp. 1–4, Dec. 2013.
[41] D. K. Tayal, A. Jain, S. Arora, S. Agarwal, T. Gupta, and N. Tyagi, "Crime detection and criminal identification in India using data mining technique," *AI Soc.*, vol. 30, no. 1, pp. 117–127, Feb. 2015.
[42] R. Kumar and B. Nagpal, "Analysis and prediction of crime patterns using big data," *Int. J. Inf. Technol.*, vol. 11, no. 4, pp. 799–805, Dec. 2019.
[43] H. K. R. ToppiReddy, B. Saini, and G. Mahajan, "Crime prediction & monitoring framework based on spatial analysis," *Proc. Comput. Sci.*, vol. 132, pp. 696–705, Jan. 2018.
[44] S. Sivaranjani, S. Sivakumari, and M. Aasha, "Crime prediction and forecasting in Tamilnadu using clustering approaches," in *Proc. Int. Conf. Emerg. Technological Trends (ICETT)*, Oct. 2016, pp. 1–6.
[45] V. Pednekar, T. Mahale, P. Gadhve, and A. Gore, "Crime Rate Prediction using KNN," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 6, no. 1, pp. 124–127, Jan. 2018.
[46] *San Francisco Crime Data*. Accessed: Jul. 12, 2020. [Online]. Available: https://data.sfgov.org/Public-Safety/Police-Department-Incident-Reports-Historical-2003/tmnf-yvry
[47] *Chicago Crime Data*. Accessed: Jul. 12, 2020. [Online]. Available: https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present-Dashboard/5cd6-ry5g
[48] *Weather Data*. Accessed: Jul. 12, 2020. [Online]. Available: https://api.weather.com/v1/location/KSFO:9: U.S./observations/historical.json?apiKey=6532d6454b8aa370768e63d6ba5a832e&units=m&startDate=
[49] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2009.
[50] P. Deepan, "Fusion of deep learning models for improving classification accuracy of remote sensing images," *J. Mech. Continua Math. Sci.*, vol. 14, no. 5, pp. 189–201, Oct. 2019.
[51] R. L. Graham, A. C. Yao, and F. F. Yao, "Information bounds are weak in the shortest distance problem," *J. ACM*, vol. 27, no. 3, pp. 428–444, Jul. 1980.
[52] P. Cortez, "Sensitivity analysis for time lag selection to forecast seasonal time series using neural networks and support vector machines," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2010, pp. 1–8.

NOWSHIN TASNIM was born in Chattogram, Bangladesh. She is currently pursuing the B.Sc. degree in computer science and engineering with the Khulna University of Engineering and Technology (KUET), Bangladesh. She has participated in various Olympiads and achieved reputable ranks in the regional Mathematical Olympiad and Physics Olympiad. She has published one poster paper on her bio-medical work based on deep learning methods. Her current research interests include data mining, image processing, machine learning models, deep learning models, and bio-medical work based on deep learning techniques.

IFTEKHER TOUFIQUE IMAM is currently pursuing the B.Sc. degree in computer science and engineering with the Khulna University of Engineering and Technology (KUET), Bangladesh. He is a contest programmer and participated in various onsite programming contests, such as ICPC Dhaka Regional 2019, 2020, and NCPC 2019. He also participates in online programming contests hosted in Codeforces and Codechef and gained reputable ratings for solving over 1500 programming problems in various online judge. He has good experience in software development and published a paper about his work on document verification with blockchain. His current research interests include blockchain, computer graphics, machine learning, data mining, and deep learning.

M. M. A. HASHEM received the bachelor's degree in electrical and electronic engineering from the Khulna University of Engineering and Technology (KUET), Khulna, Bangladesh, in 1988, the master's degree in computer science from the Asian Institute of Technology (AIT), Bangkok, Thailand, in 1993, and the Ph.D. degree in artificial intelligence systems from Saga University, Japan, in 1999. He is currently a Professor with the Department of Computer Science and Engineering, KUET. He has published more than 100 refereed articles in international journals and conference papers. He has coauthored a book *Evolutionary Computations: New Algorithms and their Applications to Evolutionary Robots, Series: Studies in Fuzziness and Soft Computing* (Berlin/New York: Springer-Verlag, 2004). He also designed and implemented a country-wide Research and Educational Network (REN) called BdREN under a project of the Ministry of Education, Bangladesh, funded by World Bank. His research interests include artificial intelligence, machine learning, data analytics, bioinformatics, health informatics, biomedical instrumentation, mobile computing, and soft computing.