Comparative Analysis of LSTM Architectures for Crime occurrence time prediction

Khawla Hasan Almahjoub
Computer Science Department,
Libyan Academy, Tripoli, Libya
K.Al-Mahjoub@uot.edu.ly

Abduelbaset Mustafa Goweder
Computer Science Department
Libyan Academy, Tripoli, Libya
agoweder@academy.edu.ly

Abstract— Crime prediction has gained increasing attention due to the growing availability of historical crime data and the need for data-driven decision-making in public safety. This study presents a comparative analysis of Long Short-Term Memory (LSTM) architectures for predicting the exact occurrence time of crimes based on temporal patterns. Three LSTM-based models are evaluated: Vanilla LSTM, Stacked LSTM, and Bidirectional LSTM.

The proposed approach integrates time-based features and lag features to capture temporal dependencies within crime data. Model performance is assessed using standard regression metrics, including Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). Experimental results indicate that deeper LSTM architectures combined with temporal lag information improve prediction accuracy compared to the baseline model.

This study demonstrates the effectiveness of LSTM-based models for crime occurrence prediction and provides insights into selecting suitable deep learning architectures for time-series crime analysis, supporting the development of more reliable tools for proactive crime prevention.

Keywords— Crime Prediction; Deep Learning; LSTM; Time Series Forecasting; Crime Data Analysis.

I. INTRODUCTION

Crime remains one of the most pressing challenges faced by modern societies, posing threats to public safety, social stability, and economic development. Accurate prediction of when crimes are likely to occur is a key step toward enabling law enforcement agencies to allocate resources more effectively and take preventive measures in advance [1].

Recent advancements in artificial intelligence and deep learning have opened new opportunities for analyzing large-scale crime datasets. Unlike traditional statistical approaches, deep learning models such as Long Short-Term Memory (LSTM) networks are specifically designed to capture temporal dependencies, making them highly effective in forecasting time-dependent events [2].

This paper focuses on the comparative evaluation of three LSTM-based models—Vanilla LSTM, Stacked LSTM, and Bidirectional LSTM—for predicting the exact date of crime occurrences. The primary objective is to determine which architecture provides the most accurate predictions, thereby contributing to the development of more reliable crime forecasting systems.

II. BACKGROUND

This section provides an overview of the fundamental concepts necessary to understand the models used in this study. It briefly explains the role of deep learning techniques in sequence modeling. In particular, it highlights the Long Short-Term Memory (LSTM) architecture, its key components, and how it addresses challenges such as vanishing gradients in traditional recurrent neural networks. This background serves as a foundation for understanding the comparative analysis conducted in this research.

Crime prediction is an interdisciplinary research area that combines criminology, social sciences, and computer science. The main objective is to leverage historical crime records to forecast future crime occurrences and provide decision support for law enforcement agencies. Traditional time series approaches such as ARIMA and other statistical models have been widely used to model temporal patterns in crime data [3]. However, these approaches are often limited in capturing complex nonlinear dependencies.

With the advancement of deep learning, Recurrent Neural Networks (RNNs) were introduced to handle sequential data by preserving information across time steps. Nevertheless, standard RNNs suffer from the

DOI:10.5281/zenodo.18073392

vanishing gradient problem, which restricts their ability to model long-term dependencies [4]. To overcome this limitation, Long Short-Term Memory (LSTM) networks were proposed to overcome the limitations of traditional RNNs, particularly the vanishing gradient problem. LSTMs extend RNNs by introducing a memory cell and gating mechanisms (the input, forget, and output gates) which allow them to retain and control relevant information over long sequences [5].

As shown in Figure 1, which is adopted from Vidhya [6], the architecture of an LSTM cell demonstrates how these gates interact to regulate the flow of information. The forget gate decides which parts of the previous state should be discarded, the input gate updates the cell with new information, and the output gate determines the information passed to the next time step. This structure enables LSTMs to effectively capture both short-term and long-term dependencies in sequential data, making them highly suitable for time-series prediction tasks such as crime forecasting.

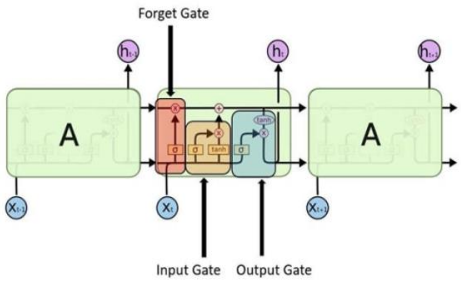

Figure 1. LSTM architecture

Moreover, variations of LSTM architectures have been developed to improve predictive power. Stacked LSTM refers to multiple LSTM layers stacked on top of each other, enabling the model to learn hierarchical temporal features. Bidirectional LSTM (BiLSTM) processes the sequence in both forward and backward directions, capturing past and future dependencies simultaneously [7]. These architectures have been widely applied in domains such as speech recognition, healthcare, and crime forecasting, showing superior performance in capturing temporal dynamics compared to traditional models [4]. These architectures show promising results in handling temporal crime patterns.

As shown in Figure 2, inspired by Nama [8], the illustration was modified and redesigned by the authors to represent the three main LSTM architectures: Vanilla LSTM, Stacked LSTM, and Bidirectional LSTM.

The Vanilla LSTM is the simplest form of Long Short-Term Memory network, consisting of a single LSTM layer followed by a dense output layer. It captures temporal dependencies in one direction (from past to future) and is effective for basic time-series forecasting tasks [4].

The Stacked LSTM extends the Vanilla architecture by adding multiple LSTM layers on top of each other. This hierarchical structure enables the model to learn more complex and abstract temporal patterns by allowing deeper feature extraction from the sequential data [7].

The Bidirectional LSTM (BiLSTM) further enhances the learning process by processing the input sequence in both forward and backward directions. This allows the model to capture dependencies from both past and future contexts, which is beneficial when the entire sequence is available and contextual understanding improves prediction accuracy [4].

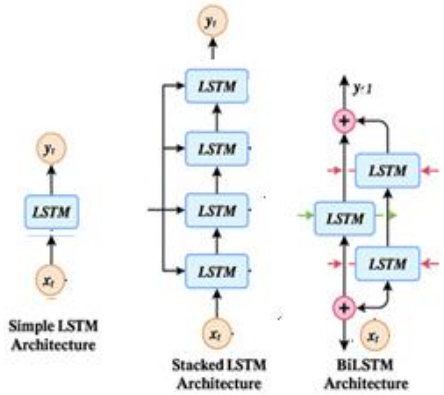

Figure 2. Architectures of Vanilla LSTM, Stacked LSTM, and Bidirectional LSTM

III. LITERATURE REVIEW

This section reviews existing research relevant to crime prediction using machine learning and deep learning techniques.

A. Machine learning Techniques

Crime prediction has been implemented in various applications. The first research paper, that utilized the same dataset employed in our research study, is titled *"Crime Analysis Through Machine Learning"* by Suhong Kim et al. [9]. Their study used two datasets: crime data from Vancouver's open data catalog (including crime type, time, and location) and neighborhood boundaries. Two algorithms were applied: K-Nearest Neighbor (KNN) and Boosted Decision Tree. The findings indicated that the Boosted Decision Tree achieved an accuracy of 43.2%, outperforming the KNN, which attained an accuracy of 41.9%.

Difference: Our research study employs deep learning models, specifically LSTM variants, to effectively

DOI:10.5281/zenodo.18073392

capture temporal patterns that are not efficiently addressed by traditional machine learning models.

Zubi and Mahmmud [10] conducted a study in Libya entitled *"Using data mining techniques to analyze crime patterns in the Libyan national crime data."* This study used manually collected crime data from police departments in Libya. The Simple K-Means algorithm was applied for clustering and Apriori for association rule mining. The outcomes suggested that two clusters were mainly associated with age, revealing higher crime rates observed in younger individuals.
Difference: Unlike their limited dataset and clustering focus, our research study uses larger datasets and applies predictive deep learning algorithms (specifically LSTM) for time-based forecasting.

Mahmud, Nuha, and Sattar [11] conducted a study entitled *"Crime Rate Prediction Using Machine Learning and Data Mining."* Over a period of three years, they gathered a dataset from Bangladesh, which was pre-processed to incorporate features like gender, age, and month. The algorithms that were used included KNN, Naïve Bayes, and Linear Regression, with KNN attaining an accuracy of 76.93%.
Difference: While their study focused on demographic classification, our research study emphasizes sequential time-series prediction using LSTM architectures.

B. Deep learning Techniques

Stec and Klabjan [12] conducted a study utilizing deep neural networks to predict daily crime counts across city grids in Chicago and Portland. Crime data was combined with weather, census, and transport data. This dataset was modeled with the application of RNN, CNN, and hybrid RNN–CNN architectures. The accuracy rates were recorded at 75.6% for Chicago and 65.3% for Portland.
Difference: Their work predicted daily aggregated counts, whereas our research study aims to predict specific date of crime occurrences.

Devi and Kavitha [13] developed a crime forecasting model based on the N-Beats algorithm, which integrates Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM). The study utilized crime data from Sacramento (2014–2021), The N-Beats model was implemented to predict future crime counts by capturing temporal patterns from historical data. Model performance was evaluated using Mean Absolute Error (MAE) and Symmetric Mean Absolute Percentage Error (SMAPE), achieving a low MAE value of 0.1407, indicating strong predictive accuracy.
Difference: While their model predicts overall crime trends, the methodology we employ seeks to predict specific date of crime occurrences.

Finally, Safat et al. [14] conducted a comprehensive study to evaluate the effectiveness of machine learning (ML) and deep learning (DL) approaches for crime classification and forecasting. They utilized large, publicly available Chicago crime data (2001–2019) for classification tasks and Los Angeles crime data (2010–2018) for time series forecasting. The study benchmarked several models, including Logistic Regression, Random Forest, XGBoost, LSTM, and ARIMA, they assessed their models using evaluation metrics, such as: accuracy, precision, recall, MAE, and RMSE. Their results demonstrated that XGBoost achieved the highest accuracy (94.00%) for crime classification and hotspot identification, surpassing all other ML models. For time series forecasting, the LSTM model outperformed the traditional ARIMA model, achieving an RMSE of 12.66 and MAE of 11.70 for the Chicago dataset, and an RMSE of 8.78 and MAE of 6.00 for the Los Angeles dataset.
Difference: They compared ML and DL models; our study focuses on optimizing LSTM architectures (Vanilla, Stacked, and Bidirectional) for enhanced temporal prediction.

IV. DATA PREPARATION

This section describes the dataset used to train and test the three main LSTM architectures, including data pre-processing and feature engineering techniques which are applied to enhance the quality and usability of the data.

A. Dataset Description

Historical crime data, including details such as crime type, date, and neighborhood information, is downloaded from Kaggle website which is available at https://www.kaggle.com/datasets/wosaku/crime-in-vancouver.
The dataset contains attributes such as *type, year, month, day, hour, minute, hundred_block, neighbourhood, x, y, latitude, and longitude.* Each instance of the dataset is a crime record with date and timestamp. The dataset is covering the period from 2003 to 2017 and comprising approximately 530,653 crime records.

B. Preprocessing

In the preprocessing phase, the dataset was cleaned to ensure its reliability and consistency. Duplicated and missing records were identified and removed, resulting in a refined dataset suitable for further analysis. This step was essential to eliminate noise that could negatively affect the training process and model performance.

To illustrate the pre-processing phase, Figures 3 and 4 show the dataset information before and after cleaning respectively. The initial dataset contained duplicated and missing records. For instance, the 'TYPE' attribute

DOI:10.5281/zenodo.18073392

originally had 530,652 records, of which 474,014 were non-null. After cleaning and removing incomplete records, the dataset became more consistent and ready for further analysis, with its size reduced to 474,014 entries. This step ensured that only complete and valid records were retained for subsequent feature engineering and model training.

```
] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 530652 entries, 0 to 530651
Data columns (total 12 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   TYPE            530652 non-null  object
 1   YEAR            530652 non-null  int64
 2   MONTH           530652 non-null  int64
 3   DAY             530652 non-null  int64
 4   HOUR            476290 non-null  float64
 5   MINUTE          476290 non-null  float64
 6   HUNDRED_BLOCK   530639 non-null  object
 7   NEIGHBOURHOOD   474028 non-null  object
 8   X               530652 non-null  float64
 9   Y               530652 non-null  float64
 10  Latitude        530652 non-null  float64
 11  Longitude       530652 non-null  float64
dtypes: float64(6), int64(3), object(3)
memory usage: 48.6+ MB
```

Figure 3: Dataset information before preprocessing.

```
] df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 474014 entries, 0 to 530651
Data columns (total 12 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   TYPE            474014 non-null  object
 1   YEAR            474014 non-null  int64
 2   MONTH           474014 non-null  int64
 3   DAY             474014 non-null  int64
 4   HOUR            474014 non-null  float64
 5   MINUTE          474014 non-null  float64
 6   HUNDRED_BLOCK   474014 non-null  object
 7   NEIGHBOURHOOD   474014 non-null  object
 8   X               474014 non-null  float64
 9   Y               474014 non-null  float64
 10  Latitude        474014 non-null  float64
 11  Longitude       474014 non-null  float64
dtypes: float64(6), int64(3), object(3)
memory usage: 47.0+ MB
```

Figure 4: Dataset information after preprocessing.

C. Feature Engineering

After preprocessing, a set of temporal features was created to capture patterns relevant to crime occurrence. The engineered features, include:

TIMESTAMP: we generate date column by combining the individual attributes of year, month, day, and hour into a single date column was converted into a Unix timestamp (seconds since January 1, 1970) to allow for numerical modeling.

DAY_OF_WEEK: Extracted from the timestamp to represent the weekday of the crime event.

IS_WEEKEND: A binary feature that indicates whether the incident occurred on Saturday or Sunday.

IS_HOLIDAY: Generated using the Python holidays library to mark whether the incident occurred on a public holiday.

D. Lag Feature Engineering

Lag features are temporal attributes that represent the values of a variable at previous time steps, allowing the model to capture short-term dependencies in the data. For this study, lag features were created from the crime occurrence timestamps by shifting the time series to generate Lag-1, Lag-2, and Lag-3 features. These features provide the model with historical context, enabling it to identify temporal patterns that influence future crime events.

An illustration of lag feature generation is shown in *Table I* where the original timestamp column is expanded to include three lagged versions (Lag 1, Lag 2, and Lag 3). For any given row, the Lag 1 feature contains the timestamp from the preceding row, Lag 2 contains the timestamp from two rows prior, and so on. These lagged values allow the model to learn from past events, which is particularly important in time-series forecasting where past occurrences strongly affect future outcomes.

The NaN (Not a Number) values appear in the initial rows because no preceding data exists for the first few entries in the time series. For example, the first row has no prior data, so all its lag features are NaN. These missing values are an expected outcome of the lagging process and are typically handled by removing the affected rows

Table I: Lag feature generation from TIMESTAMP

TIMESTAMP	Lag 1	Lag 2	Lag 3
2017-03-01 10:36	NaN	NaN	NaN
2017-03-03 21:50	2017-03-01 10:36	NaN	NaN
2017-03-04 06:12	2017-03-03 21:50	2017-03-01 10:36	NaN
2017-03-07 14:47	2017-03-04 06:12	2017-03-03 21:50	2017-03-01 10:36
2017-03-08 03:14	2017-03-07 14:47	2017-03-04 06:12	2017-03-03 21:50

In practice, lag features were integrated only in the Stacked LSTM and Bidirectional LSTM models. This decision was motivated by the need to enhance these models' ability to capture sequential dependencies over time.

V. MODEL BUILDING

In this section, we implemented three deep learning models to forecast crime occurrences based on temporal features, these are:
Vanilla LSTM, Stacked LSTM, and Bidirectional LSTM models.

1) Vanilla LSTM Model

As illustrated in Figure 5, the Vanilla LSTM Diagram consists of a single LSTM layer with 150 units, this number was chosen after experimenting with commonly used values such as 100, 150, and 200. It is

DOI:10.5281/zenodo.18073392

observed that 150 units offered the best trade-off between model accuracy and computational efficiency. The single LSTM layer is followed by a dropout layer (40% of the units are dropped to reduce overfitting). Similarly, the 40% dropout rate was selected after testing multiple rates (e.g., 20%, 30%, 40%, and 50%). It was found that a 40% dropout provided the optimal balance between preventing overfitting and maintaining sufficient learning capacity. Lower dropout values did not sufficiently reduce overfitting, while higher ones caused the model to lose important temporal information during training. Finally, a dense output layer for timestamp prediction. This baseline architecture captures temporal dependencies without introducing additional complexity.

During model building, the dataset was divided into training and validation subsets using an 80/20 split ratio to ensure a fair evaluation of the model's generalization capability. The model was trained using the Adam optimizer and Mean Squared Error (MSE) as the loss function, as it is well-suited for continuous value prediction tasks. Training was performed over multiple epochs until the validation loss stabilized, indicating that the model had effectively learned temporal crime patterns from the historical data.

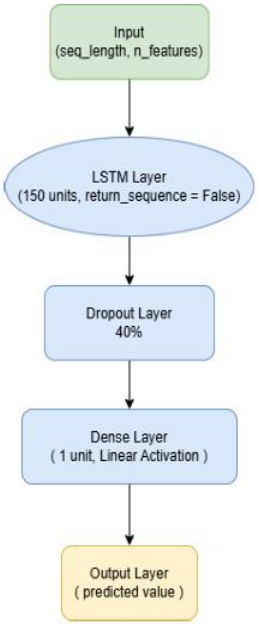

Figure 5 Vanilla LSTM model Diagram.

2) Stacked LSTM Model

The Stacked LSTM model includes two LSTM layers stacked on top of each other to capture deeper temporal dependencies. Lag features were additionally introduced at the input stage to enrich the representation of short-term patterns. A dropout layer and dense output layer complete the Diagram. Figure 6 illustrates the Diagram of the Stacked LSTM model.

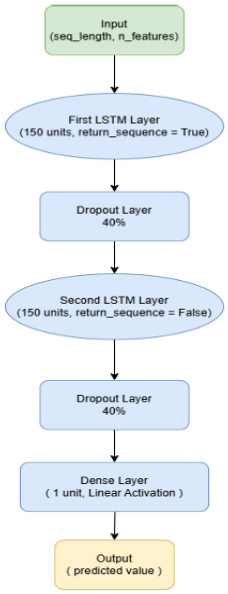

Figure 6 Stacked LSTM model Diagram.

3) Bidirectional LSTM Model

Figure 7 illustrates the Diagram of the Bidirectional LSTM model. The Bidirectional LSTM model processes the input sequence in both forward and backward directions, allowing the model to capture temporal dependencies from past and future contexts simultaneously. Similar to the stacked model, lag features were also introduced in this architecture to enhance accuracy

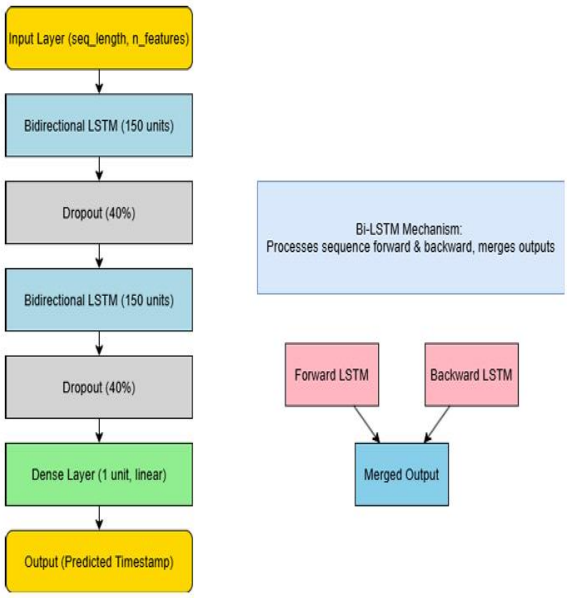

Figure 7 Bidirectional LSTM model Diagram.

DOI:10.5281/zenodo.18073392

4) Specific Crime–Neighborhood LSTM Model

The Specific LSTM model extends the Stacked LSTM architecture based on its superior performance in crime occurrence time prediction. By focusing on the most frequent crime type within the most frequently reported neighborhood, the model reduces data heterogeneity and enables more focused temporal learning. The same preprocessing, feature engineering, and lag feature construction steps applied in the previous models are retained to capture short-term temporal dependencies. The network architecture itself remains identical to that of the Stacked LSTM model; the only distinction is an initial data filtering step performed prior to training, where the dataset is restricted to the most frequent crime type in the most frequently reported neighborhood.

VI. MODELS' *RESULTS AND PERFORMANCE*

This section discusses the performance evaluation concerning the three LSTM-based models used for predicting crime dates. These three models were built and tested. The performance of each model was measured using three standard evaluation metrics — Mean Squared Error (MSE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE). These metrics, which are specifically used to assess the built models, are discussed.

Table II: A summary of the obtained results for each model.

Measure / Model	MSE	MAE	RMSE
Vanilla LSTM	0.0003809739	0.01654	0.01951
Stacked LSTM (with Lag Features)	0.0000016893	0.00104	0.00129
Bidirectional LSTM (with Lag Features)	0.0000017653	0.00156	0.00132
Specific Model	0.0000045435	0.00168	0.00213

As shown in Table II, the Stacked LSTM model with lag features achieved the best overall performance among all developed models. It recorded the lowest error values across all evaluation metrics, with MSE = 0.0000016893, MAE = 0.0010452796, and RMSE = 0.00129, indicating its superior capability in capturing temporal dependencies and producing highly accurate crime occurrence time predictions.

The Bidirectional LSTM model with lag features produced slightly higher error values (MSE = 0.0000017653, MAE = 0.0015660838, RMSE = 0.00132). This suggests that processing sequences in both forward and backward directions did not yield additional performance benefits for this strictly time-dependent prediction task.

The Specific Model, which focuses on the most frequent crime type within the most frequently reported neighborhood, achieved competitive performance with MSE = 0.0000045435, MAE = 0.00168, and RMSE = 0.00213. Although its error values are slightly higher than those of the general Stacked LSTM model, the results demonstrate that targeted data filtering can still provide reliable and more focused predictions, offering practical value for localized crime analysis.

Finally, the Vanilla LSTM model recorded the highest error rates (MSE = 0.0003809739, MAE = 0.0165465735, RMSE = 0.01951), confirming that deeper architectures and the incorporation of lag features play a crucial role in enhancing prediction accuracy.

VII. CONCLUSIONS

This study highlights the effectiveness of LSTM-based deep learning models for crime occurrence time prediction. The experimental results demonstrate that incorporating lag features and adopting deeper architectures significantly improves predictive accuracy. Specifically, the Stacked LSTM model achieved lower error values compared to both Vanilla and Bidirectional LSTM models across all evaluation metrics with an MSE of 0.00000168, an RMSE of 0.00129, and an MAE of 0.00104. Although the Bidirectional LSTM captures temporal dependencies from both forward and backward directions, it did not outperform the Stacked LSTM in this time-dependent prediction task.

Furthermore,The proposed Specific Model, which focuses on the most frequent crime type within the most frequently reported neighborhood, achieved an MSE of 0.00000454, an RMSE of 0.00213, and an MAE of 0.00168. Although these error values are slightly higher than those of the general Stacked LSTM model, they remain sufficiently low to support reliable and practically actionable predictions.

These numerical results confirm that feature engineering, model depth, and problem-specific data filtering play a critical role in enhancing prediction performance. Moreover, the use of consistent evaluation metrics (MSE, RMSE, and MAE) ensures a fair and transparent comparison across all developed models.

VIII. FUTURE WORK

Future research could focus on integrating external contextual data, such as weather conditions, population

DOI:10.5281/zenodo.18073392

density, or socio-economic indicators, to further improve prediction accuracy. In addition, exploring hybrid deep learning models that combine LSTM with attention mechanisms or spatial–temporal networks may capture more complex relationships between time and location. Finally, deploying these models in real-time prediction systems could provide valuable support for law enforcement.

IX. ACKNOWLEDGMENT

First and foremost, we praise and thank Allah Almighty for enabling us to complete this research study.
We would like to extend our heartfelt appreciation to the Libyan Academy and the Libyan Ministry of Higher Education and Scientific Research for their logistical assistance and support during this research study.

REFERENCES

[1] N. Shah, N. Bhagat, and M. Shah, "Crime forecasting: a machine learning and computer vision approach to crime prediction and prevention," Visual Computing for Industry, Biomedicine, and Art, vol. 4, no. 1, p. 9, 2021.

[2] M. Manengadan, S. Nandanan, and N. Subash, "Crime data analysis, visualization and prediction using LSTM," 2021.

[3] A. Stec and D. Klabjan, "Forecasting crime with deep learning," arXiv preprint arXiv:1806.01486, 2018.

[4] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "The performance of LSTM and BiLSTM in forecasting time series," in 2019 IEEE International Conference on Big Data (Big Data), 2019, pp. 3285–3292.

[5] GeeksforGeeks, "Deep Learning: Introduction to Long Short-Term Memory," [Online]. Available: https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/

[6] Analytics Vidhya, "Fundamentals of Deep Learning: Introduction to LSTM," 2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of deep-learning-introduction-to-lstm/

[7] A. Sahar and D. Han, "An LSTM-based indoor positioning method using Wi-Fi signals," in Proc. 2nd Int. Conf. on Vision, Image and Signal Processing, 2018, pp. 1–5.

[8] A. Nama, "Understanding Bidirectional LSTM for Sequential Data Processing," Medium, [Online]. Available: https://medium.com/@anishnama20/understanding-bidirectional-lstm-for-sequential-data-processing-b83d6283befc

[9] S. Kim, P. Joshi, P. S. Kalsi, and P. Taheri, "Crime analysis through machine learning," in 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2018, pp. 415–420.

[10] Z. S. Zubi and A. A. Mahmmud, "Using data mining techniques to analyze crime patterns in the libyan national crime data," in Proc. 1st WSEAS Int. Conf. Image Processing and Pattern Recognition, 2013, pp. 79-85.

[11] N. Mahmud, M. Nuha, and A. Sattar, "Crime rate prediction using machine learning and data mining," International Journal of Computer Applications, vol. 183, no. 10, pp. 36–42, 2021.

[12] A. Stec and D. Klabjan, "Forecasting crime with deep learning," arXiv preprint arXiv:1806.01486, 2018

[13] J. V. Devi and K. S. Kavitha, "Automating Time Series Forecasting on Crime Data using RNN-LSTM," International Journal of Advanced Computer Science and Applications, vol. 124 12, no. 10, 2021. A. Saxena, "An Introduction to Convolutional Neural Networks," International Journal of Research in Applied Science and Engineering Technology, vol. 10, no. 12, pp. 943–947, 2022.

[14] 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.

DOI:10.5281/zenodo.18073392