software

MDPI

Article

Deep-SDM: A Unified Computational Framework for Sequential Data Modeling Using Deep Learning Models

Nawa Raj Pokhrel [1], Keshab Raj Dahal [2,*], Ramchandra Rimal [3], Hum Nath Bhandari [4] and Binod Rimal [5]

[1] Department of Physics and Computer Science, Xavier University of Louisiana, New Orleans, LA 70125, USA; npokhrel@xula.edu
[2] Department of Mathematics, State University of New York Cortland, Cortland, NY 13045, USA
[3] Department of Mathematical Sciences, Middle Tennessee State University, Murfreesboro, TN 37132, USA; ramchandra.rimal@mtsu.edu
[4] Department of Mathematics and Physics, Roger Williams University, Bristol, RI 02809, USA; hbhandari@rwu.edu
[5] Department of Mathematics, The University of Tampa, Tampa, FL 33606, USA; brimal@ut.edu
* Correspondence: keshabraj.dahal@cortland.edu

Abstract: Deep-SDM is a unified layer framework built on TensorFlow/Keras and written in Python 3.12. The framework aligns with the modular engineering principles for the design and development strategy. Transparency, reproducibility, and recombinability are the framework's primary design criteria. The platform can extract valuable insights from numerical and text data and utilize them to predict future values by implementing long short-term memory (LSTM), gated recurrent unit (GRU), and convolution neural network (CNN). Its end-to-end machine learning pipeline involves a sequence of tasks, including data exploration, input preparation, model construction, hyperparameter tuning, performance evaluations, visualization of results, and statistical analysis. The complete process is systematic and carefully organized, from data import to model selection, encapsulating it into a unified whole. The multiple subroutines work together to provide a user-friendly and conducive pipeline that is easy to use. We utilized the Deep-SDM framework to predict the Nepal Stock Exchange (NEPSE) index to validate its reproducibility and robustness and observed impressive results.

Keywords: deep learning; sequential data modeling; time series; LSTM; GRU; CNN

Citation: Pokhrel, N.R.; Dahal, K.R.; Rimal, R.; Bhandari, H.N.; Rimal, B. Deep-SDM: A Unified Computational Framework for Sequential Data Modeling Using Deep Learning Models. *Software* **2024**, *3*, 47–61. https://doi.org/10.3390/software3010003

Academic Editor: Claus Pahl

Received: 7 December 2023
Revised: 23 February 2024
Accepted: 26 February 2024
Published: 28 February 2024

Highlights

- A simplified and scalable framework for implementing deep learning model architectures: LSTM, GRU, and CNN for sequential data modeling.
- Extensive, automated, and data-driven approach for hyperparameter tuning.
- Numerous subroutines for data exploration, input preparation, model construction, hyperparameter tuning, statistical validation, and results visualization.
- Tracks every intermediate and final result.
- Implements robust model selection strategies and performs statistical analysis to validate the conclusions.

1. Introduction

Many powerful computational frameworks have been released recently for prediction and classification domains [1–9]. Interestingly, these developed frameworks are guided by different schools of thought, such as regression, traditional machine learning, and deep learning. However, regardless of the approach, machine learning practitioners, researchers, engineers, and data scientists often have to assemble the functionalities from multiple frameworks to solve their unique problems. This gap has inspired the development of a unified layered computational architecture that integrates end-to-end activities, enabling complete machine learning pipelines. The developed layered architecture falls under a

deep learning framework that combines basic functionalities with statistical validation, making it a valuable tool for sequential modeling and time series forecasting.

Deep learning architectures that can process vast amounts of data, recognize patterns, and make accurate predictions have opened up new possibilities across various sectors, leading to increased efficiency, improved decision-making, and enhanced user experiences. It has revolutionized many industries, including manufacturing [10–12], finance [13,14], healthcare [15–18], environment [19], electronics [20], energy [21,22], agriculture [23,24], transportation [25,26], entertainment [27,28], retail [29,30], e-commerce [31,32], and many others, transforming the way we approach complex tasks and unlocking new possibilities. Although it is a relatively new and emerging technology, many data-driven or rule-based algorithms, from naive to complex, are already employed in various scientific fields [6,33–39]. With few exceptions, most of the architecture uses structured numerical data to predict the given subject area [40–43]. The existing machine learning framework lacks an integrated framework for combining numerical and text data for prediction. The proposed Deep-SDM architecture provides two separate modules for processing the numerical and text inputs and then combines the outputs from these modules in a subsequent layer. As a result, the model can learn from mixed data types and capture the relationship in a broader spectrum. The cooperative framework is versatile and can handle multiple machine learning tasks, including data collection, wrangling, and numerical and text data preprocessing. Additionally, it integrates model construction, algorithmic implementation, and statistical validation. These activities are integrated into a holistic framework, making the prediction process more efficient and streamlined.

The Deep-SDM architecture employs a machine learning pipeline that consists of a four-layered architecture. Figure 1 presents the schematic diagram of the architecture from an aerial perspective. A pipeline aims to organize and automate the steps to create an end-to-end workflow for developing and deploying deep learning architectures. Each layer has a specific responsibility and interacts with the layers above and below it in a controlled manner. **Layer 1** consists of two branches that independently handle numerical and text data, both of which are time series data with the date as an index. These two datasets are ultimately merged into a single data frame, yielding a multivariate time series as the model input. **Layer 2** is responsible for preprocessing and normalizing the concatenated data to prepare the input sequence for the model. **Layer 3** provides several tools to construct and implement three different deep learning architectures—LSTM, GRU, and CNN with multiple configurations. Users can customize the model configurations with the desired complexity depending on the underlying real-world problem while implementing the selected architecture. The optimal model architecture is identified by utilizing extensive hyperparameter tuning strategies. **Layer 4** provides several auxiliary subroutines to evaluate and visualize the model outcomes and perform the statistical analysis.

Transparency, reproducibility, and recombinability are the framework's primary design criteria. Therefore, all the source codes are available on GitHub (https://github.com/mlrg2020/Deep-SDM/blob/main/Deep-SDM.ipynb (accessed on 25 February 2024)). Additionally, Deep-SDM is characterized by the following key features:

- **Scalable Framework:** It is a simplified and scalable framework for implementing deep learning model architectures: LSTM, GRU, and CNN for sequential data modelling.
- **Unified Pipeline:** The framework handles multiple machine learning tasks, such as data exploration, input preparation, model construction, hyperparameter tuning, statistical validation, and results visualization. These activities are integrated together to make a unified framework.
- **Optimization Methodology:** It provides an extensive, automated, and data-driven approach for hyperparameter tuning to select the optimal parameter of the model.
- **Ease of Use:** Every intermediate and final result is accessed through the objects or can be stored in a separate file for future purposes.
- **Model Validation:** The framework implements robust model selection strategies and performs statistical analysis to validate the conclusions. Root Mean Squared Error

(RMSE), Mean Absolute Percentage Error (MAPE), and Correlation Coefficient (R) are the evaluation metrics for model selection.

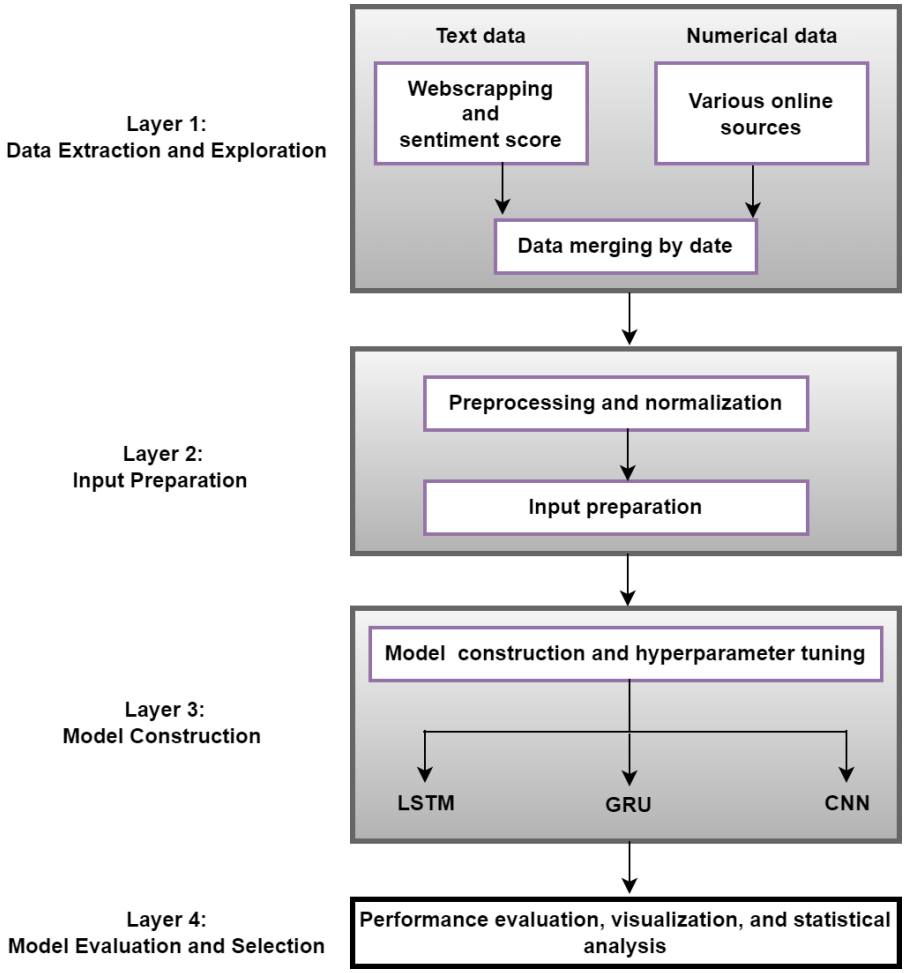


Figure 1. Analytical framework of the Deep-SDM architecture.

The Deep-SDM framework has been used in a recently published research article to predict the closing price of the stock market index [44] and observed the impressive result. We have presented the major outcomes in Appendix A, an illustrative example demonstrating the framework's reproducibility. Moreover, the developed pipeline can be customized and expanded to include additional stages or specialized techniques based on the project's requirements.

The lack of maturity in the subject area has resulted in a scarcity of previously published research articles on the integrated computational framework. Existing research studies often speak about the reliability of their framework on their own method. However, it is important to note that the model framework, underlying assumptions, and implementation vary significantly from one study to another. Thus, it is not easy to compare published research articles unbiasedly, even if they use the same deep learning architecture to construct their predictive models.

The rest of the paper is organized as follows: Section 2 explains the related work in this field. Section 3 introduces and explains LSTM, CNN, and GRU along with their functionalities for implementation. It also explores the code metadata and computing environment. Section 4 explains the implementation of the proposed framework together with illustrative examples. Section 5 discusses the impact of the architecture on related fields. We have presented the discussion in Section 6. Finally, Section 7 presents conclusions, followed by acknowledgments and a relevant list of references. Appendix A is presented to justify our model architecture further at the end.

2. Related Work

Over the last few years, studies on developing a time series forecasting framework have been growing rapidly. Both academic researchers and practitioners have made tremendous efforts in this field. This section focuses on the review of previous studies on the frameworks built for time series forecasting.

In 2020, Livieris et al. proposed a framework that ensures a time series is suitable for fitting a deep learning model by performing a series of transformations to satisfy the stationarity property. The enforcement of stationarity is performed by the application of the Augmented Dickey–Fuller test, and the deep learning model's predictions are guaranteed by rejecting the hypothesis of autocorrelation in the model's errors using the Ljung–Box Q test. They used data from the financial market, energy sector, and cryptocurrency areas. The performance of forecasting models was compared. The empirical method indicated that their proposed methodology considerably improves the forecasting performance of the deep learning models [45]. Similarly, in 2017, Bao et al. proposed a deep learning framework where wavelet transforms, stacked autoencoders, and LSTM were combined for time series forecasting. The time series data were decomposed by wavelet transformation to eliminate noise, then deep high-level features were generated by autoencoders, and, finally, extracted features were fed into LSTM to forecast the next day's closing price of six market indices. They claim that the proposed model outperforms similar models in predictive accuracy and profitability performance [3]. Furthermore, in 2020, Yan et al. proposed a deep learning framework that integrated complementary ensemble empirical mode decomposition, principal component analysis, and LSTM to predict the closing price of the next trading day. The author claims that their proposed framework outperforms benchmark models in both absolute and risk-adjusted profitability performance on six different datasets, including the New York Stock Exchange, Dow Jones Index, and S&P 500 Index [4].

In 2019, Chen and Shi proposed a deep learning framework using a relative position matrix and convolutional neural network for the time series classification task. They investigated a time series data representation method called relative position matrix to convert the raw time series data to two-dimensional images, which enables the use of techniques from image recognition. They also constructed an improved convolutional neural network architecture to automatically learn a high-level abstract representation of low-level raw time series data. The author claims that their proposed framework outperformed the existing time series classification models by a significant margin [46].

In 2018, Du and Horing [47] proposed a sequence-to-sequence deep learning framework for multivariate time series forecasting to address the dynamic, spatial–temporal, and nonlinear characteristics of multivariate time series data using LSTM-based encoder–decoder architecture. Through the air quality multivariate time series forecasting experiments, they claimed that their proposed model had better forecasting performance than classic shallow learning and baseline deep learning models. In 2019, Du et al. proposed a hybrid deep learning architecture to extract local trends, spatial correlation, and spatial–temporal dependencies from multivariate time series data. The first component of the architecture consists of layers of a one-dimensional CNN component. The concatenated output from CNN is then fed into Bi-LSTM layers, which predict the targeted sequence value at the end. The experimental results on two datasets show that the computational framework outperforms the baseline statistical/machine learning models (ARIMA, SVM) and deep learning models (LSTM, GRU, CNN) based on evaluation metrics RMSE and MAE [5]. Similarly, Khorram et al. utilized a CNN–LSTM hybrid classification model architecture for fault diagnosis [48], while Gilik et al. employed a similar computational framework by combining two/three-dimensional CNN and LSTM architecture for air quality prediction [7].

In 2022, Wang et al. proposed a novel framework for interpretable deep learning in time series analysis by embedding a human–machine collaborative knowledge representation within an autoencoder architecture [49]. The framework addresses the "black-box"

nature of deep learning models. Experimental results across various datasets demonstrate competitive classification performance while ensuring interpretability. Additionally, integrating a human-in-the-loop mechanism further improves classification accuracy by allowing human intervention, offering a promising framework for sequential data modeling and time series analysis.

DeepAD [50], an anomaly detection framework for time series data, leverages various time series forecasting models to enhance anomaly detection accuracy without relying on labeled datasets for training or optimizing thresholds based on class labels. To test the framework's efficiency, the authors performed comparative experiments against the EGADS framework [51] on real and synthetic data and demonstrated significant performance improvements. The study concludes by highlighting the potential for further enhancement by integrating additional time series forecasting models, such as convolutional neural networks, particularly for spatiotemporal datasets.

The paper by Li et al. introduces a deep learning framework coupled with time series analysis methods for runoff prediction in the middle Yangtze River [52]. The framework enhances the precision of daily runoff prediction compared to existing models by incorporating LSTM networks and two-time series methods, namely, mutual information and seasonal and trend decomposition using LOESS. The key findings of the work include the framework's adaptability to data quantity analysis, robustness in capturing seasonal trends, and minimal prediction errors. This work demonstrates the potential of combining deep learning with time series analysis for more accurate runoff prediction in hydrology research.

Yao et al. proposed a computational framework for sequence modeling focusing on proteins, which comprises two independent feature extraction techniques: a pre-trained Bert-encoder and a 1D-CNN + LSTM. The first is responsible for the feature representation of the sequence, while the 1D-CNN + LSTM model captures the compositional, evolutionary, and physicochemical properties of peptides. These two sets of features are then concatenated and fed into a feed-forward network to classify two groups of peptides. Experimental results demonstrate a high accuracy of over 92% across three different datasets [53].

Somu et al. developed a hybrid computational framework incorporating k-means clustering, CNN, and LSTM. The first component aims to discern sequence patterns/trends, the second extracts complex features with non-linear interactions, and the third captures long-term dependencies by modeling temporal information in time series data. The model's robustness is evaluated using an energy consumption dataset [8].

Yang et al. [54] proposed a deep learning framework that employs the LSTM model as a benchmark to predict the financial indicators' values. They used three financial indicators to validate their proposed model: return on tangible assets, price-to-sales ratio, and price-earnings ratio of Mondelez International and Hormel Food Corp stock companies. They claim that their proposed framework is of great importance to forecasting the financial risks in the financial sector, regardless of the size of an organization.

Aiming to capture the nonlinearity inherent in multivariate financial time series, the author [55] proposed a deep learning framework that integrates the two-stage feature selection model and the deep learning with the error correction model. Case studies and the corresponding sensitivity analysis were carried out to validate the performance of their forecasting framework. Based on these studies, they claim that their proposed framework outperformed the existing sixteen benchmark models.

In 2021, Wen and Yang [9] designed a framework with three learning parts, namely, LSTM based on the Generate Performance Model, ensemble learning based on the Restrict and Control Model, and one-dimensional CNN of the Dirichlet distribution based on the Overall Verification Model. Before learning steps, they exploit the K-means method as pre-processing and hypothesis verification to improve the prediction accuracy. After learning the steps, they constructed four forecasting progresses—the Point by Point Generated Method, Sequence Full Generated Method, Sequence Multiple Generated Method, and Improvement with Restrict and Control Model and Overall Verification Model—to predict the

positions and trends of dimensions in each system. The author claims that their proposed framework performs better than that of the former authors.

Ye and Dai introduced TrEnOS-ELMK, a hybrid algorithm blending transfer learning and ensemble learning. It leverages long-ago data for improved time series forecasting. Their approach combines an Online Sequential Extreme Learning Machine with Kernels. Comparative evaluations against numerous existing methods across synthetic and real-world datasets showcased its superior performance [56].

The studies mentioned above have delved into various methodologies proposed to address the complexities inherent in sequence modeling. Each approach showed its unique blend of techniques to the forefront and expanded the horizons of sequence forecasting by highlighting the potential for cross-pollination of ideas across domains. These advancements underscore a dynamic landscape, wherein researchers continually push the envelope to refine existing techniques and explore novel paradigms in pursuit of more accurate and interpretable sequence forecasting methodologies.

3. Methods and Functionalities

The proposed framework focuses on implementing LSTM, GRU, and CNN models for various sequential modeling tasks. LSTM is an improved version of a vanilla recurrent neural network (RNN) designed to capture long-term dependencies in sequential data. It consists of input, hidden state, cell state, and output, with four gates controlling the flow of information [57]. LSTM maintains a memory cell to retain information over long sequences, making it suitable for time series forecasting and natural language processing tasks. GRU is a variation of LSTM that simplifies the architecture by merging the hidden and cell states into a single vector [58]. GRU has three gates controlling information flow, is computationally more efficient than LSTM, and is quite popular for sequential data analysis. LSTM and GRU have been successfully applied to solve problems in various domains, such as finance [59], ecology [60,61], medical imaging analysis [62,63], and natural language processing [64,65].

CNN was initially developed for image processing and has been adapted for sequential data analysis [66]. In the context of sequential data, CNNs treat each time step as a one-dimensional image and perform convolutional operations over time. CNNs are effective for capturing spatial patterns in sequential data and have been successful in tasks like image recognition [67], cybersecurity [68], object detection [69], sentiment analysis [70], and medical diagnosis [71].

The overall functionality of Deep-SDM is illustrated in Figure 2, which can be summarized into four categories associated with the four layers: (a) data extraction and exploration, (b) input preparation, (c) model construction, and (d) model evaluation and selection.

In the first stage, various functions access numerical and text data from the intended sources. Functions such as *read_csv*() from Pandas library [72] are utilized to read the numerical data, while unstructured data are cleaned up using functions such as *word_tockenize*(), *punctuation*(), and *WordNetLemmmatizer*() extracted from the *nltk* library [73]. Furthermore, we rely on the *SentimentIntensityAnalyzer*() that uses the VEDER package [74] to calculate sentiment scores of the unstructured data. These scores are combined with numerical data through the *merge*() function by the index date.

The ultimate goal of the framework is sequential data prediction, irrespective of any subject area. Thus, the second stage prepares the input sequence and its corresponding output based on the time step or look-back period. A *DataCreation*() function is constructed, which returns (input, output) pairs when time steps are supplied. By providing (input, output) pairs based on the given time step, we can move forward with *data_split*() to partition the original data into train–validation (or train–test) subsets. These subsets are normalized using *min_max_transformation*() for optimal performance. Later in the process, the model prediction is inverse-transformed back into the original scale using *min_max_inverse_transform*().

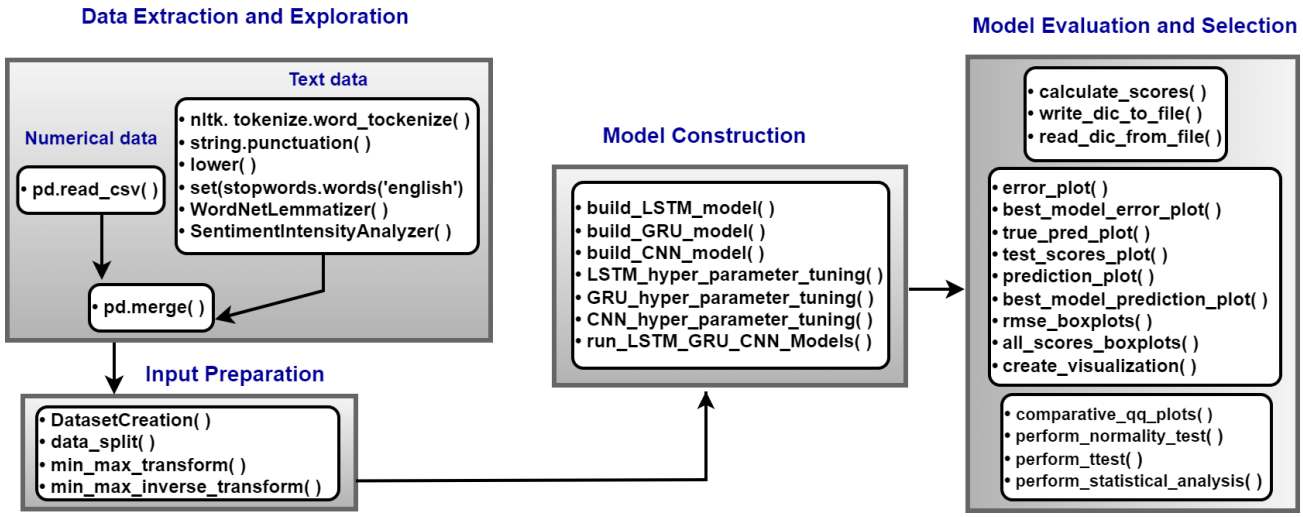


Figure 2. Workflow analysis of Deep-SDM functionalities.

The framework's third stage includes functionalities that allow the creation of deep learning models like LSTM, GRU, and CNN. These models can be customized with several neurons/filters, layers, and other hyperparameters like learning rate, batch size, filter size, optimization method, etc. The Tensorflow and Keras APIs are used through the *build_LSTM_model*(), *build_GRU_model*(), and *build_CNN_model*() subroutines. Users can experiment with the hyperparameters of the deep learning models using the *LSTM_hyper_parameter_tuning*(), *GRU_hyper_parameter_tuning*(), and *CNN_hyper_parameter_tuning*() functions, which are automated and data-driven with available validation data. The *run_LS-TM_GRU_CNN_models*() function automates the execution of multiple/single-layer LSTM/GRU/CNN models, with inputs such as data, hyperparameters, time step, test split, epochs, number of replicates, and provides complete performance scores on both train and test data as an output dictionary. The crucial inputs like the time step, number of features, optimizer, batch size, and learning rate can also be easily customized in these routines.

When carrying out a computational project, it is essential to consider model evaluation and selection as the last stage. Once the results are available, it is vital to calculate model performance scores, visualize them, and conduct statistical analysis. The *calculate_scores*() function computes the standard metrics, such as RMSE, MAPE, and R. The RMSE is calculated as

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2},$$

which measures the average deviation of predicted values from the actual values. The MAPE measures the percentage deviation of predicted values from the actual values and is computed using the following formula.

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\%$$

The R measures the strength and direction of the linear relationship between predicted and actual values and is obtained using the formula given below.

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

where,

y_i: True value;

$\bar{y_i}$: The average value of the original sequence;

$\hat{y_i}$: Predicted value;

$\bar{\hat{y}_i}$: Average value of predicted sequence;

n: Number of observations.

The results visualization and statistical analysis section provides various functions that enable users to visualize results and perform statistical tests efficiently. By using a single driver routine called *create_visualization*(), which combines different subroutines listed in Figure 2, users can obtain prediction plots, error plots, and performance score plots. Furthermore, the *perform_statistical_analysis*() function uses subroutines designed for carrying out normality tests and two-sample t-tests, which help to validate conclusions drawn from other empirical experiments.

Table 1 summarizes the code metadata and the computational framework of the Deep-SDM framework. The brief information on several attributes of code metadata, such as the current code version, permanent link to the reproducible capsule, and legal code license, is presented in Table 1.

Table 1. Code metadata and environmental setup.

Info: Code Metadata and Environmental Setup	**Description**
Current code version	*Version 1.0*
Permanent link to code/repository used for this code version	https://github.com/mlrg2020/Deep-SDM
Permanent link to Reproducible Capsule	https://codeocean.com/capsule/3410053/tree/v1
Legal Code License	*MIT License*
Compilation requirements, operating environments, and dependencies	Code can be executed in Anaconda or Google Colab
Support email for questions	ml.researchgroup2020@gmail.com
Machine Configuration	Google Colab with NVIDIA-SMI 495.44 GPU
Environment	Python 3.6.0, TensorFlow, and Keras APIs
Architecture	LSTM, GRU, and CNN

4. Implementation of the Proposed Framework: Illustrative Example

As part of our effort to analyze the robustness of the proposed model, we applied the Deep-SDM framework to predict financial market indices, explicitly focusing on the NEPSE index, in a study presented in [44]. The research objective was to forecast the closing price of the NEPSE index for the following day using multivariate input sequences. We developed several deep learning models based on LSTM, GRU, and CNN architectures utilizing selected input variables and evaluated their performances using standard assessment metrics: RMSE, MAPE, and R. The overall strategy of the Deep-SDM implementation and the experimental results are provided in Figures A1–A6, and Tables A1 and A2 in the Appendix A.

A broad overview of the Deep-SDM model building and implementation strategy can be found in Figure A1, whereas the process of data exploration of the response variable (i.e., closing price) and input preparation are presented in Figure A2. Similarly, Figures A3–A5 provide the key experimental results, including the average performance scores of constructed models, true vs. predicted values plot, and true vs. predicted time series in test data. The optimal set of hyperparameters for each model architecture is presented in Table A1. Similarly, Table A2 provides the list of the best performing models with their corresponding hyperparameters. Finally, Figure A6 provides a normality plot of RMSE distributions of the best performing models obtained from multiple replicates. For more detailed information, readers are encouraged to visit the full paper [44].

5. Impact on Related Fields

This study aims to enhance research in deep learning and sequential data modeling. It provides developers with powerful tools and comprehensive frameworks, allowing easy experimentation with new models and techniques. The automated machine learning pipelines comprise pre-built modules and functions so developers can focus on the application level instead of detailed implementation technicalities. It leads to quick prototyping and deployment of deep learning models.

One of the significant advantages of this study is its open-source code that follows a collaborative framework. It can democratize deep learning by enabling a broader range of activities to benefit from deep learning models, including those without extensive programming or mathematical expertise. The developed framework can open up opportunities for newcomers, promote knowledge-sharing and collaboration, and provide consistent implementation and well-documented procedures for researchers and practitioners to reproduce and validate results, fostering transparency and reliability. It can also positively impact education by providing learning resources, tutorials, and examples that facilitate the understanding of and application of deep learning concepts. Ultimately, companies and researchers dealing with sequential data may use this package to analyze data and make informed decisions.

6. Discussion

Modeling sequential data presents significant challenges due to its variable input length, long-term dependencies, and interrelationship with other factors. Despite extensive efforts [5,24,43,75,76] to create an adaptive framework, variations in time windows and input features across studies make it difficult to have a unified framework. In addition, the differences in model architectures, their configurations, and implementation complexity make direct comparisons challenging. Addressing these challenges requires a computational framework capable of capturing multifaceted information in an identical environment. Additionally, ensuring consistency in the computational environment and transparently addressing ethical considerations are crucial aspects of model implementation.

The article delves into a promising development in deep learning architecture, which can have significant implications for predictive modeling. We can predict subject areas in time series data by utilizing LSTM, GRU, and CNN models in a unified deep learning framework. The architecture is designed to handle diverse input features, including textual and numerical data, to capture the variability often found in mixed data types. Our study follows established guidelines for modular engineering paradigms to create a reliable and effective framework for deep learning architecture. The developed framework has the potential to unlock the capabilities of deep learning, enabling better predictions and helping to make more informed decisions across various disciplines.

7. Conclusions

Deep-SDM aims to develop a unified computational framework that gathers information from various sources. It provides clear guidelines for data preprocessing, model building, hyperparameter tuning, model evaluation, visualization, model comparison, and statistical validation. This integrated computing framework is designed for supervised sequential data modeling, offering user-friendly functionalities.

The developed machine learning pipeline helps to automate the process of building and deploying LSTM, GRU, and CNN models under the integrated compuational environment. The outcome of the pipeline ensures validity, reliability, consistency, and reproducibility, instilling a sense of trust and confidence in the architecture. The machine learning scripts are also meticulously inspected to ensure interpretability, regardless of the domain. As a result, it enhances the usability of the models and enables the users to make well-informed decisions. Additionally, the model's outcome can be considered as an additional piece of information to delineate the cone of uncertainty in the given subject area.

In the near future, we plan to compare our framework with existing literature to enhance the trustworthiness further. Another potential direction is to implement the existing framework with Transformers and Generative Adversarial Networks. We also plan to enhance the framework with a sub-routine that leverages Intent Recognition and Topic Modeling techniques on text data to extract meaningful features, thereby augmenting the predictive power of the framework. The enhancements aim to refine and extend the capabilities of the Deep-SDM framework to meet evolving challenges in sequential data modeling from an alternative perspective.

Author Contributions: N.R.P.: Conceptualization, Methodology, Validation, Formal analysis, Investigation, Writing—original draft, Writing—review & editing. K.R.D.: Conceptualization, Methodology, Validation, Software, Visualization, Investigation, Writing—original draft, Writing—review & editing. R.R.: Methodology, Validation, Visualization, Investigation, Writing—original draft, Writing—review & editing. H.N.B.: Methodology, Validation, Visualization, Software, Writing—original draft, Writing—review & editing. B.R.: Methodology, Validation, Visualization, Software, Writing—original draft, Writing—review & editing. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data are publicly available on Codeocean and Github. The detailed information is provided in Table 1.

Acknowledgments: The authors would like to thank the Association of Nepalese Mathematicians in America for creating a collaborative research opportunity that resulted in this work.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A

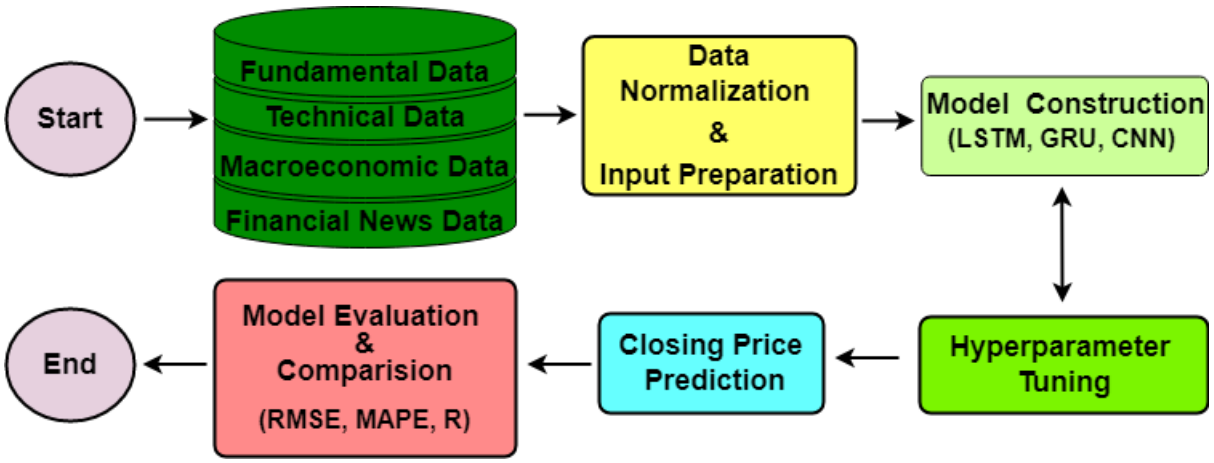
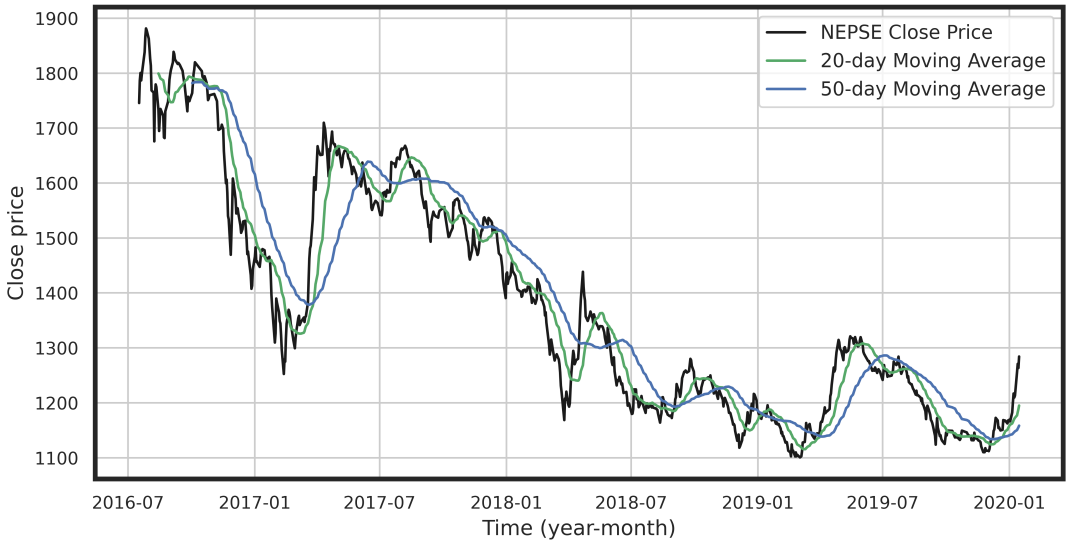
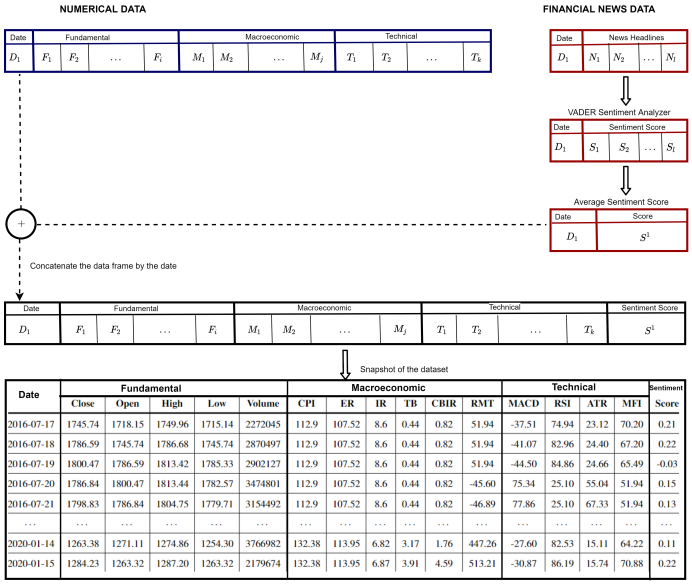


Figure A1. Model building and implementation strategy.





Date	Fundamental					Macroeconomic						Technical				Sentiment
	Close	Open	High	Low	Volume	CPI	ER	IR	TB	CBIR	RMT	MACD	RSI	ATR	MFI	Score
2016-07-17	1745.74	1718.15	1749.96	1715.14	2272045	112.9	107.52	8.6	0.44	0.82	51.94	-37.51	74.94	23.12	70.20	0.21
2016-07-18	1786.59	1745.74	1786.68	1745.74	2870497	112.9	107.52	8.6	0.44	0.82	51.94	-41.07	82.96	24.40	67.20	0.22
2016-07-19	1800.47	1786.59	1813.42	1785.33	2902127	112.9	107.52	8.6	0.44	0.82	51.94	-44.50	84.86	24.66	65.49	-0.03
2016-07-20	1786.84	1800.47	1813.44	1782.57	3474801	112.9	107.52	8.6	0.44	0.82	-45.60	75.34	25.10	55.04	51.94	0.15
2016-07-21	1798.83	1786.84	1804.75	1779.71	3154492	112.9	107.52	8.6	0.44	0.82	-46.89	77.86	25.10	67.33	51.94	0.13
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2020-01-14	1263.38	1271.11	1274.86	1254.30	3766982	132.38	113.95	6.82	3.17	1.76	447.26	-27.60	82.53	15.11	64.22	0.11
2020-01-15	1284.23	1263.32	1287.20	1263.32	2179674	132.38	113.95	6.87	3.91	4.59	513.21	-30.87	86.19	15.74	70.88	0.22

Figure A2. Data exploration (response variable trend) and data preparation.

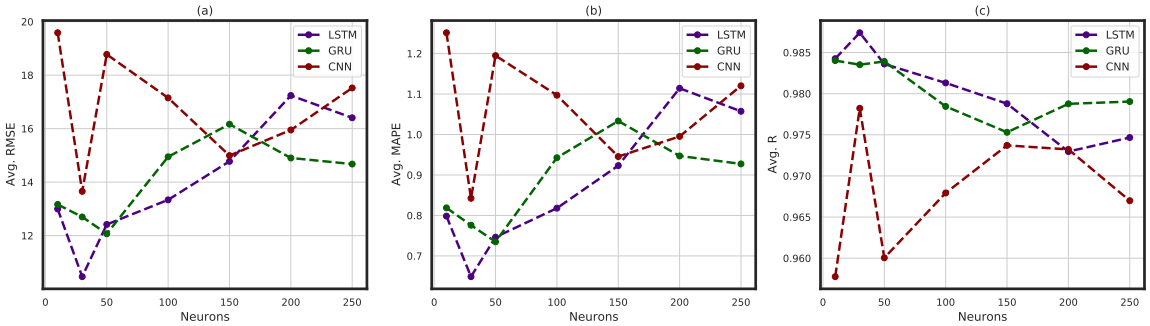


Figure A3. (**a**–**c**) Average performance scores of different models.

Table A1. List of the best hyperparameters for the LSTM, GRU, and CNN models.

No. of Neurons/Filters	LSTM			GRU			CNN		
	Optimizer	Learning Rate	Batch Size	Optimizer	Learning Rate	Batch Size	Optimizer	Learning Rate	Batch Size
10	Adam	0.01	16	Adam	0.001	4	Adagrad	0.01	8
30	Adam	0.1	16	Adagrad	0.01	8	Adagrad	0.1	8
50	Adam	0.001	16	Adagrad	0.01	16	Adagrad	0.01	16
100	Adagrad	0.01	16	Adagrad	0.001	8	Adagrad	0.01	16
150	Adagrad	0.001	4	Adagrad	0.001	16	Adagrad	0.01	16
200	Adagrad	0.001	16	Adagrad	0.001	16	Adagrad	0.01	16
250	Adagrad	0.001	16	Adagrad	0.001	16	Adagrad	0.001	8

Table A2. Best performing LSTM, GRU, and CNN models with their best hyperparameters.

Models	No. of Neurons	Optimizer	Learning Rate	Batch Size
LSTM	30	Adam	0.1	16
GRU	50	Adagrad	0.01	16
CNN	30	Adagrad	0.1	8

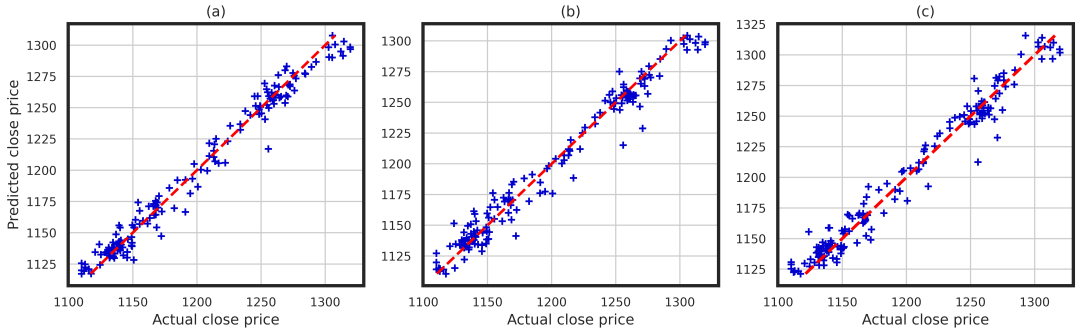


Figure A4. (**a**–**c**) True vs. predicted values in the test data.

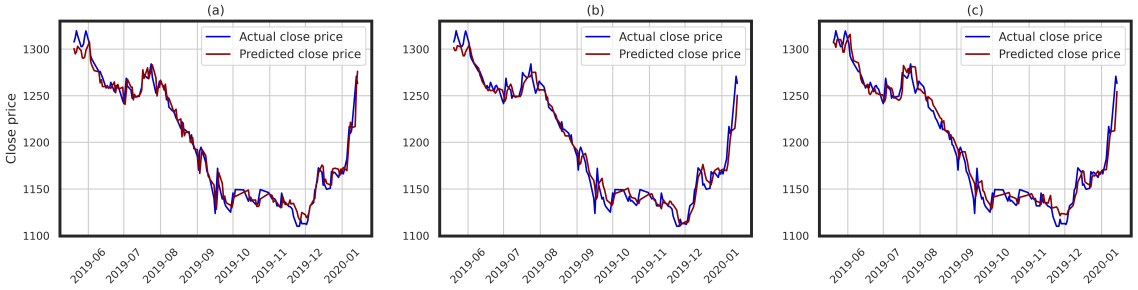


Figure A5. (**a**–**c**) True and predicted time series plots in the test data.

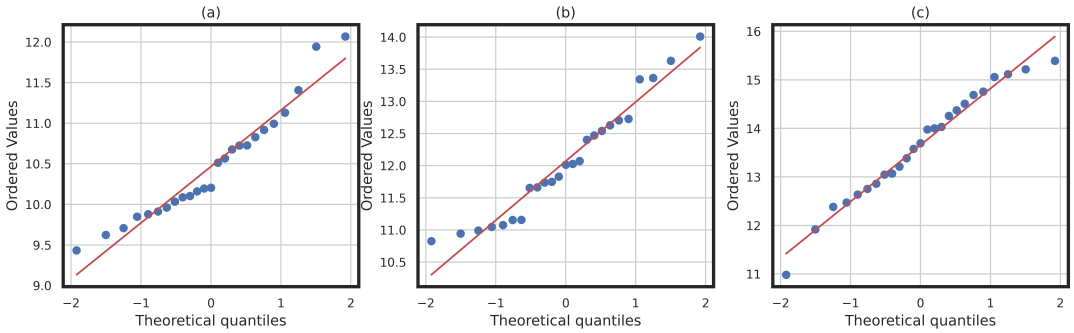


Figure A6. (**a**–**c**) Normality plots of RMSEs.

References

1. Alkhatib, K.; Khazaleh, H.; Alkhazaleh, H.A.; Alsoud, A.R.; Abualigah, L. A new stock price forecasting method using active deep learning approach. *J. Open Innov. Technol. Mark. Complex.* **2022**, *8*, 96. [CrossRef]
2. Mandal, M.; Vipparthi, S.K. An empirical review of deep learning frameworks for change detection: Model design, experimental frameworks, challenges and research needs. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 6101–6122. [CrossRef]
3. Bao, W.; Yue, J.; Rao, Y. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLoS ONE* **2017**, *12*, e0180944. [CrossRef] [PubMed]
4. Yan, B.; Aasma, M. A novel deep learning framework: Prediction and analysis of financial time series using CEEMD and LSTM. *Expert Syst. Appl.* **2020**, *159*, 113609.
5. Du, S.; Li, T.; Yang, Y.; Horng, S.J. Deep air quality forecasting using hybrid deep learning framework. *IEEE Trans. Knowl. Data Eng.* **2019**, *33*, 2412–2424. [CrossRef]
6. Dahal, K.R.; Pokhrel, N.R.; Gaire, S.; Mahatara, S.; Joshi, R.P.; Gupta, A.; Banjade, H.R.; Joshi, J. A comparative study on effect of news sentiment on stock price prediction with deep learning architecture. *PLoS ONE* **2023**, *18*, e0284695. [CrossRef] [PubMed]
7. Gilik, A.; Ogrenci, A.S.; Ozmen, A. Air quality prediction using CNN+ LSTM-based hybrid deep learning architecture. *Environ. Sci. Pollut. Res.* **2022**, *29*, 11920–11938. [CrossRef]
8. Somu, N.; MR, G.R.; Ramamritham, K. A deep learning framework for building energy consumption forecast. *Renew. Sustain. Energy Rev.* **2021**, *137*, 110591. [CrossRef]
9. Wen, S.C.; Yang, C.H. Time series analysis and prediction of nonlinear systems with ensemble learning framework applied to deep learning neural networks. *Inf. Sci.* **2021**, *572*, 167–181. [CrossRef]
10. Yang, J.; Li, S.; Wang, Z.; Dong, H.; Wang, J.; Tang, S. Using deep learning to detect defects in manufacturing: A comprehensive survey and current challenges. *Materials* **2020**, *13*, 5755. [CrossRef]
11. Malhan, R.; Gupta, S.K. The Role of Deep Learning in Manufacturing Applications: Challenges and Opportunities. *J. Comput. Inf. Sci. Eng.* **2023**, *23*, 060816. [CrossRef]
12. Jamwal, A.; Agrawal, R.; Sharma, M. Deep learning for manufacturing sustainability: Models, applications in Industry 4.0 and implications. *Int. J. Inf. Manag. Data Insights* **2022**, *2*, 100107. [CrossRef]
13. Sahu, S.K.; Mokhade, A.; Bokde, N.D. An Overview of Machine Learning, Deep Learning, and Reinforcement Learning-Based Techniques in Quantitative Finance: Recent Progress and Challenges. *Appl. Sci.* **2023**, *13*, 1956. [CrossRef]
14. Huang, J.; Chai, J.; Cho, S. Deep learning in finance and banking: A literature review and classification. *Front. Bus. Res. China* **2020**, *14*, 1–24. [CrossRef]
15. Abdel-Jaber, H.; Devassy, D.; Al Salam, A.; Hidaytallah, L.; El-Amir, M. A review of deep learning algorithms and their applications in healthcare. *Algorithms* **2022**, *15*, 71. [CrossRef]
16. Kaul, D.; Raju, H.; Tripathy, B. Deep learning in healthcare. In *Deep Learning in Data Analytics: Recent Techniques, Practices and Applications*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 97–115.
17. Othman, N.A.; Abdel-Fattah, M.A.; Ali, A.T. A Hybrid Deep Learning Framework with Decision-Level Fusion for Breast Cancer Survival Prediction. *Big Data Cogn. Comput.* **2023**, *7*, 50. [CrossRef]
18. Roy, B.; Malviya, L.; Kumar, R.; Mal, S.; Kumar, A.; Bhowmik, T.; Hu, J.W. Hybrid Deep Learning Approach for Stress Detection Using Decomposed EEG Signals. *Diagnostics* **2023**, *13*, 1936. [CrossRef] [PubMed]
19. Ko, K.K.; Jung, E.S. Improving Air Pollution Prediction System through Multimodal Deep Learning Model Optimization. *Appl. Sci.* **2022**, *12*, 10405. [CrossRef]
20. Kothadiya, D.; Bhatt, C.; Sapariya, K.; Patel, K.; Gil-González, A.B.; Corchado, J.M. Deepsign: Sign language detection and recognition using deep learning. *Electronics* **2022**, *11*, 1780. [CrossRef]
21. Forootan, M.M.; Larki, I.; Zahedi, R.; Ahmadi, A. Machine learning and deep learning in energy systems: A review. *Sustainability* **2022**, *14*, 4832. [CrossRef]
22. Olu-Ajayi, R.; Alaka, H.; Sulaimon, I.; Sunmola, F.; Ajayi, S. Building energy consumption prediction for residential buildings using deep learning and other machine learning techniques. *J. Build. Eng.* **2022**, *45*, 103406. [CrossRef]
23. Altalak, M.; Ammad uddin, M.; Alajmi, A.; Rizg, A. Smart agriculture applications using deep learning technologies: A survey. *Appl. Sci.* **2022**, *12*, 5919. [CrossRef]
24. Islam, M.M.; Adil, M.A.A.; Talukder, M.A.; Ahamed, M.K.U.; Uddin, M.A.; Hasan, M.K.; Sharmin, S.; Rahman, M.M.; Debnath, S.K. DeepCrop: Deep learning-based crop disease prediction with web application. *J. Agric. Food Res.* **2023**, *14*, 100764. [CrossRef]
25. Nagaraj, N.; Gururaj, H.L.; Swathi, B.H.; Hu, Y.C. Passenger flow prediction in bus transportation system using deep learning. *Multimed. Tools Appl.* **2022**, *81*, 12519–12542. [CrossRef] [PubMed]
26. Ravi, C.; Tigga, A.; Reddy, G.T.; Hakak, S.; Alazab, M. Driver identification using optimized deep learning model in smart transportation. *ACM Trans. Internet Technol.* **2022**, *22*, 84. [CrossRef]
27. Sayal, A.; Chaithra, N.; Jha, J.; Trilochan, B.; Kalyan, G.V.; Priya, M.S.; Gupta, V.; Memoria, M.; Gupta, A. Visual Sentiment Analysis Using Machine Learning for Entertainment Applications. In Proceedings of the 2023 International Conference on IoT, Communication and Automation Technology (ICICAT), Gorakhpur, India, 23–24 June 2023; pp. 1–8.
28. Chou, C.H.; Su, Y.S.; Hsu, C.J.; Lee, K.C.; Han, P.H. Design of desktop audiovisual entertainment system with deep learning and haptic sensations. *Symmetry* **2020**, *12*, 1718. [CrossRef]

29. Nasseri, M.; Falatouri, T.; Brandtner, P.; Darbanian, F. Applying Machine Learning in Retail Demand Prediction—A Comparison of Tree-Based Ensembles and Long Short-Term Memory-Based Deep Learning. *Appl. Sci.* **2023**, *13*, 11112. [CrossRef]
30. Giri, C.; Chen, Y. Deep learning for demand forecasting in the fashion and apparel retail industry. *Forecasting* **2022**, *4*, 565–581. [CrossRef]
31. Zhang, X.; Guo, F.; Chen, T.; Pan, L.; Beliakov, G.; Wu, J. A Brief Survey of Machine Learning and Deep Learning Techniques for E-Commerce Research. *J. Theor. Appl. Electron. Commer. Res.* **2023**, *18*, 2188–2216. [CrossRef]
32. Alzahrani, M.E.; Aldhyani, T.H.; Alsubari, S.N.; Althobaiti, M.M.; Fahad, A. Developing an intelligent system with deep learning algorithms for sentiment analysis of E-commerce product reviews. *Comput. Intell. Neurosci.* **2022**, *2022*, 3840071. [CrossRef]
33. Bhandari, H.N.; Rimal, B.; Pokhrel, N.R.; Rimal, R.; Dahal, K.R.; Khatri, R.K. Predicting stock market index using LSTM. *Mach. Learn. Appl.* **2022**, *9*, 100320. [CrossRef]
34. Chen, H.; Engkvist, O.; Wang, Y.; Olivecrona, M.; Blaschke, T. The rise of deep learning in drug discovery. *Drug Discov. Today* **2018**, *23*, 1241–1250. [CrossRef]
35. Fan, C.; Sun, Y.; Zhao, Y.; Song, M.; Wang, J. Deep learning-based feature engineering methods for improved building energy prediction. *Appl. Energy* **2019**, *240*, 35–45. [CrossRef]
36. He, Y.; Wu, P.; Li, Y.; Wang, Y.; Tao, F.; Wang, Y. A generic energy prediction model of machine tools using deep learning algorithms. *Appl. Energy* **2020**, *275*, 115402. [CrossRef]
37. Lu, W.; Li, J.; Wang, J.; Qin, L. A CNN-BiLSTM-AM method for stock price prediction. *Neural Comput. Appl.* **2021**, *33*, 4741–4753. [CrossRef]
38. Zhu, J.; Wang, J.; Wang, X.; Gao, M.; Guo, B.; Gao, M.; Liu, J.; Yu, Y.; Wang, L.; Kong, W.; et al. Prediction of drug efficacy from transcriptional profiles with deep learning. *Nat. Biotechnol.* **2021**, *39*, 1444–1452. [CrossRef] [PubMed]
39. Sarwar, M.A.; Kamal, N.; Hamid, W.; Shah, M.A. Prediction of diabetes using machine learning algorithms in healthcare. In Proceedings of the 2018 24th International Conference on Automation and Computing (ICAC), Newcastle upon Tyne, UK, 6–7 September 2018; IEEE: New York, NY, USA, 2018; pp. 1–6.
40. Mohan, S.; Mullapudi, S.; Sammeta, S.; Vijayvergia, P.; Anastasiu, D.C. Stock price prediction using news sentiment analysis. In Proceedings of the 2019 IEEE Fifth International Conference on Big Data Computing Service and Applications (BigDataService), Newark, CA, USA, 4–9 April 2019; IEEE: New York, NY, USA, 2019; pp. 205–208.
41. Abdullah, S.S.; Rahaman, M.S.; Rahman, M.S. Analysis of stock market using text mining and natural language processing. In Proceedings of the 2013 International Conference on Informatics, Electronics and Vision (ICIEV), Dhaka, Bangladesh, 17–18 May 2013; IEEE: New York, NY, USA, 2013; pp. 1–6.
42. Schumaker, R.P.; Chen, H. Textual analysis of stock market prediction using breaking financial news: The AZFin text system. *ACM Trans. Inf. Syst. (TOIS)* **2009**, *27*, 1–19. [CrossRef]
43. Bhandari, H.N.; Rimal, B.; Pokhrel, N.R.; Rimal, R.; Dahal, K.R. LSTM-SDM: An integrated framework of LSTM implementation for sequential data modeling. *Softw. Impacts* **2022**, *14*, 100396. [CrossRef]
44. Pokhrel, N.R.; Dahal, K.R.; Rimal, R.; Bhandari, H.N.; Khatri, R.K.; Rimal, B.; Hahn, W.E. Predicting nepse index price using deep learning models. *Mach. Learn. Appl.* **2022**, *9*, 100385. [CrossRef]
45. Livieris, I.E.; Stavroyiannis, S.; Pintelas, E.; Pintelas, P. A novel validation framework to enhance deep learning models in time-series forecasting. *Neural Comput. Appl.* **2020**, *32*, 17149–17167. [CrossRef]
46. Chen, W.; Shi, K. A deep learning framework for time series classification using Relative Position Matrix and Convolutional Neural Network. *Neurocomputing* **2019**, *359*, 384–394. [CrossRef]
47. Du, S.; Li, T.; Horng, S.J. Time series forecasting using sequence-to-sequence deep learning framework. In Proceedings of the 2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), Taipei, Taiwan, 26–28 December 2018; pp. 171–176.
48. Khorram, A.; Khalooei, M.; Rezghi, M. End-to-end CNN+ LSTM deep learning approach for bearing fault diagnosis. *Appl. Intell.* **2021**, *51*, 736–751. [CrossRef]
49. Wang, J.; Li, R.; Li, R.; Fu, B.; Chen, D.Z. Hmckrautoencoder: An interpretable deep learning framework for time series analysis. *IEEE Trans. Emerg. Top. Comput.* **2022**, *10*, 99–111. [CrossRef]
50. Buda, T.S.; Caglayan, B.; Assem, H. Deepad: A generic framework based on deep learning for time series anomaly detection. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Melbourne, VIC, Australia, 3–6 June 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 577–588.
51. Laptev, N.; Amizadeh, S.; Flint, I. Generic and scalable framework for automated time-series anomaly detection. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, 10–13 August 2015; pp. 1939–1947.
52. Li, Z.; Kang, L.; Zhou, L.; Zhu, M. Deep learning framework with time series analysis methods for runoff prediction. *Water* **2021**, *13*, 575. [CrossRef]
53. Yao, L.; Zhang, Y.; Li, W.; Chung, C.R.; Guan, J.; Zhang, W.; Chiang, Y.C.; Lee, T.Y. DeepAFP: An effective computational framework for identifying antifungal peptides based on deep learning. *Protein Sci.* **2023**, *32*, e4758. [CrossRef] [PubMed]
54. Yang, S. A novel study on deep learning framework to predict and analyze the financial time series information. *Future Gener. Comput. Syst.* **2021**, *125*, 812–819. [CrossRef]

55. Niu, T.; Wang, J.; Lu, H.; Yang, W.; Du, P. Developing a deep learning framework with two-stage feature selection for multivariate financial time series forecasting. *Expert Syst. Appl.* **2020**, *148*, 113237. [CrossRef]
56. Ye, R.; Dai, Q. A novel transfer learning framework for time series forecasting. *Knowl.-Based Syst.* **2018**, *156*, 74–99. [CrossRef]
57. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
58. Chung, J.; Gülçehre, Ç.; Cho, K.; Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv* **2014**, arXiv:1412.3555.
59. Cao, J.; Li, Z.; Li, J. Financial time series forecasting model based on CEEMDAN and LSTM. *Phys. A Stat. Mech. Appl.* **2019**, *519*, 127–139. [CrossRef]
60. Capinha, C.; Ceia-Hasse, A.; Kramer, A.M.; Meijer, C. Deep learning for supervised classification of temporal data in ecology. *Ecol. Inform.* **2021**, *61*, 101252. [CrossRef]
61. Alshahrani, H.M.; Al-Wesabi, F.N.; Al Duhayyim, M.; Nemri, N.; Kadry, S.; Alqaralleh, B.A. An automated deep learning based satellite imagery analysis for ecology management. *Ecol. Inform.* **2021**, *66*, 101452. [CrossRef]
62. Li, H.; Fan, Y. Interpretable, highly accurate brain decoding of subtly distinct brain states from functional MRI using intrinsic functional networks and long short-term memory recurrent neural networks. *NeuroImage* **2019**, *202*, 116059. [CrossRef]
63. Rimal, R.; Brannon, M.; Wang, Y.; Yang, X. Comparative study of various machine learning methods on ASD classification. *Int. J. Data Sci. Anal.* **2023**, 1–15 . . [CrossRef]
64. Wang, D.; Su, J.; Yu, H. Feature extraction and analysis of natural language processing for deep learning English language. *IEEE Access* **2020**, *8*, 46335–46345. [CrossRef]
65. Lavanya, P.; Sasikala, E. Deep learning techniques on text classification using Natural language processing (NLP) in social healthcare network: A comprehensive survey. In Proceedings of the 2021 3rd International Conference on Signal Processing and Communication (ICPSC), Coimbatore, India, 13–14 May 2021; pp. 603–609.
66. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *J. Big Data* **2021**, *8*, 1–74. [CrossRef]
67. Lu, J.; Tan, L.; Jiang, H. Review on convolutional neural network (CNN) applied to plant leaf disease classification. *Agriculture* **2021**, *11*, 707. [CrossRef]
68. Mohammadpour, L.; Ling, T.C.; Liew, C.S.; Aryanfar, A. A survey of CNN-based network intrusion detection. *Appl. Sci.* **2022**, *12*, 8162. [CrossRef]
69. Dhillon, A.; Verma, G.K. Convolutional neural network: A review of models, methodologies and applications to object detection. *Prog. Artif. Intell.* **2020**, *9*, 85–112. [CrossRef]
70. Joloudari, J.H.; Hussain, S.; Nematollahi, M.A.; Bagheri, R.; Fazl, F.; Alizadehsani, R.; Lashgari, R.; Talukder, A. BERT-deep CNN: State of the art for sentiment analysis of COVID-19 tweets. *Soc. Netw. Anal. Min.* **2023**, *13*, 99. [CrossRef]
71. Marques, G.; Agarwal, D.; De la Torre Díez, I. Automated medical diagnosis of COVID-19 through EfficientNet convolutional neural network. *Appl. Soft Comput.* **2020**, *96*, 106691. [CrossRef] [PubMed]
72. McKinney, W. Pandas: A foundational Python library for data analysis and statistics. *Python High Perform. Sci. Comput.* **2011**, *14*, 1–9.
73. Bird, S. NLTK: The natural language toolkit. In Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, Sydney, NSW, Australia, 17 July 2006; pp. 69–72.
74. Hutto, C.; Gilbert, E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In Proceedings of the International AAAI Conference on Web and Social Media, Ann Arbor, MI, USA, 1–4 June 2014; Volume 8, pp. 216–225.
75. Mollenhauer, D.; Atzmueller, M. Sequential Exceptional Pattern Discovery Using Pattern-Growth: An Extensible Framework for Interpretable Machine Learning on Sequential Data. In Proceedings of the XI-ML@ KI, Bamberg, Germany, 21 September 2020.
76. Ostmeyer, J.; Cowell, L. Machine learning on sequential data using a recurrent weighted average. *Neurocomputing* **2019**, *331*, 281–288. [CrossRef]