Scientific Research Publishing

Journal of Computer and Communications, 2024, 12, 187-206
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

An Evaluation of Deep Learning Models for Classifying Time Series Individual Data Instances

Joshua A. Blaney, Suresh S. Muknahallipatna*

Department of Electrical Engineering and Computer Science, University of Wyoming, Laramie, WY, USA
Email: jblaney1@uwyo.edu, *sureshm@uwyo.edu

How to cite this paper: Blaney, J.A. and Muknahallipatna, S.S. (2024) An Evaluation of Deep Learning Models for Classifying Time Series Individual Data Instances. *Journal of Computer and Communications*, **12**, 187-206.
https://doi.org/10.4236/jcc.2024.1211014

Received: October 17, 2024
Accepted: November 24, 2024
Published: November 27, 2024

Abstract

Deep learning for time series sequence individual data instance classification can revolutionize computer assisted navigation by providing surgeons with accurate, real-time instrument locality through automatic instrument localization. This paper presents an evaluation of Deep Learning models to perform individual data instance classification of time series data. The models explored include convolution and recurrent networks, as well as state-of-the-art residual and inception architectures. The time series data used to evaluate the models consists of depth and force measurements from a drill. Four recurrent neural network models using long short-term memory and gated recurrent units, known as baseline models, and four models using 1D convolution with ResNet and Inception architectures, known as advanced models, were evaluated by determining the data instance membership of the four classes. The four classes represent four distinct regions in a bone traversed by the drill bit during a surgical procedure. First, the time series data is preprocessed, identifying the four classes or regions of the bone. Next, the paper presents a discussion of the network architecture and modifications of both the basic and advanced deep learning models, followed by the training process and hyperparameters tuning. The performance of the models was evaluated using the precision and recall performance parameters. Out of the eight models evaluated, the recurrent neural network with gated recurrent units has the best performance. The paper also demonstrates the importance of the feature depth over the feature force in classifying the data instances, followed by the effects of the imbalanced dataset on the performance of the models.

Keywords

Recurrent Neural Networks, Long Short Term Memory, Gated Recurrent Units, Residual Networks, Inception Networks

1. Introduction

Distal radius fractures are the most common type of fracture for both pediatric and elderly patients [1]. In the event of an unstable or displaced distal radius fracture, it is common to set portions of the affected bone through open reduction internal fixation (ORIF) with angular stable fixation plates and volar locking plates [2].

Orthopedic surgeons perform ORIF to set distal radius fractures by creating an incision to access the bone, drilling into the affected bone, and then attaching plates with screws to set the bone. During this procedure, the surgeon estimates the position of the drill in the patient's bone through the tactile response from the drill. Optionally, a metal plate may be used as a guard on the backside of the bone to prevent the drill from plunging into the soft tissue and tendons on the opposite side of the bone. Appropriate screw lengths are estimated using a depth gauge and can be verified through fluoroscopy.

The current method of using tactile feedback to estimate the position of the drill leads to hardware misplacement (incorrect screw sizes) and increased plunge depth [3]. Misplaced hardware can lead to complications with tendon irritation, tendon rupture, or nerve damage, all of which require hardware removal [4]. Plunging into the tissue surrounding the bone can lead to soft tissue damage, permanent disability, or life-threatening bleeding [5]. If surgeons can place the hardware more precisely while simultaneously decreasing plunge depth, the efficacy of distal radius surgery and similar orthopedic surgeries would increase.

Methods of separately improving hardware placement or decreasing plunge depth exist. Yet, technology to improve both simultaneously is still elusive. Hardware placement has been improved by using fluoroscopic views to check the positioning of each screw [6], and plunge depth has been decreased by using physical drill stops [5] or dual motor drills [7].

Without knowing the exact diameter of the bone, using fluoroscopic views, drill stops, or current dual motor drills will only ameliorate the complications common to this surgery, not prevent them. If the precise locality of the drill can be provided to the surgeon in real time, the surgeon would know at what depth they encounter and pierce the second bone wall. Subsequently, the surgeon would also know the appropriate length of each screw.

Surgical drills that have sensors such as force/pressure, depth, and torque sensors are currently being explored [7] [8], but have not been widely adopted. If these sensors can be leveraged by a computer-assisted navigation (CaN) system to perform drill localization within the bone, then surgeons will know where the drill is in the bone, including at what depths they have encountered and pierced the first and second bone walls.

CaN has been used to assist surgeons in their practice for approximately 30 years. It includes both active (computer augmented controllers, ACaN) and passive (computer augmented feedback, PCaN) implementation schemes [9]. Current ACaN systems require the construction of computer-aided design (CAD)

models of the patient prior to the operation, and close monitoring for accurate patient-to-model registration and instrument localization during the operation.

Patient-to-model registration is the process of synchronizing the patient's position with a model of the patient such that the anatomy of the patient aligns with the anatomy represented in the model. Instrument localization is the process of precisely locating and representing the position of the drill with respect to the patient and the operating region.

ACaN systems are expensive to operate [10], which precludes their use in operations where preoperative modeling yields marginal improvement in patient outcomes. The operating cost of ACaN can be reduced by performing automatic instrument localization without CAD modeling and patient-to-model registration. This is only feasible in operations where the instrument traverses material that is measurably unique, such as using a drill to operate on distal radius fractures.

To perform automatic instrument localization, the localization algorithm must be 1) temporally/spatially aware, 2) able to process individual data instances, and 3) capable of generalizing to new data. It is imperative that the algorithm is capable of generalizing from one patient to another and from one surgeon to another; because no preoperative modeling or operative registration will be performed, and each surgeon may use the device differently.

Statistical localization algorithms such as Kalman filters or simultaneous localization and mapping (SLAM) are spatially aware and able to process individual data instances. However, these algorithms do not generalize well from one distribution to another. Although the distribution of bone width is stable between patients, the distribution of force and velocity will not be stable from one surgeon to the next. Thus, statistical localization algorithms will fail with the large variance of the distribution of force and velocity associated with each surgeon.

The major shortcoming of statistical localization methods is how poorly they generalize; thus, approaches that can generalize, such as machine learning (ML), need to be investigated. The state-of-the-art ML approaches which are applicable to this localization task include kernel methods with distance functions and deep learning (DL).

Kernel methods with distance functions such as K-nearest neighbors (KNN) with dynamic time warping (DTW) are temporally/spatially aware, can process individual data instances, and may generalize well enough for this research. The major limiting factor with these models is the relationship between prediction speed and data availability. As more data becomes available, these models become slower and may not predict in real time.

Deep learning (DL) is well suited to perform automatic instrument localization when the problem is approached as a time series classification (TSC) task because it can address all three requirements listed below:

1) DL models for TSC include convolutional neural networks (CNN) and recurrent neural networks (RNN), which are both temporally/spatially aware network

architectures.

2) DL models for TSC can be used for individual data instance classification (IDIC) by adjusting the input data flow.

3) DL models are better at generalizing new data than the state-of-the-art kernel functions with distance measures.

In this work, LSTM and GRU-based baseline networks, as well as state-of-the-art TSC architectures, including residual networks and inception networks, are investigated to perform automatic instrument localization through IDIC.

The work is organized as follows: Section 2 clarifies the research problem, Section 3 reviews related research, Section 4 discusses data preprocessing for ML, Section 5 establishes the explored architectures and training procedure, Section 6 presents the findings, and Section 7 provides concluding remarks and future work.

2. Research Problem

Figure 1 provides an illustration of drilling through the bone to set a distal radius fracture and demonstrates an ideal sequence of data instances collected from a depth sensor and a force sensor while drilling through a bone. Ideal in this context means low noise and easily interpreted through visual analysis. Region 1 corresponds to drilling on bone wall 1; Region 2 corresponds to drilling between bone walls; Region 3 corresponds to drilling on bone wall 2; and Region 4 corresponds to drilling beyond the bone.

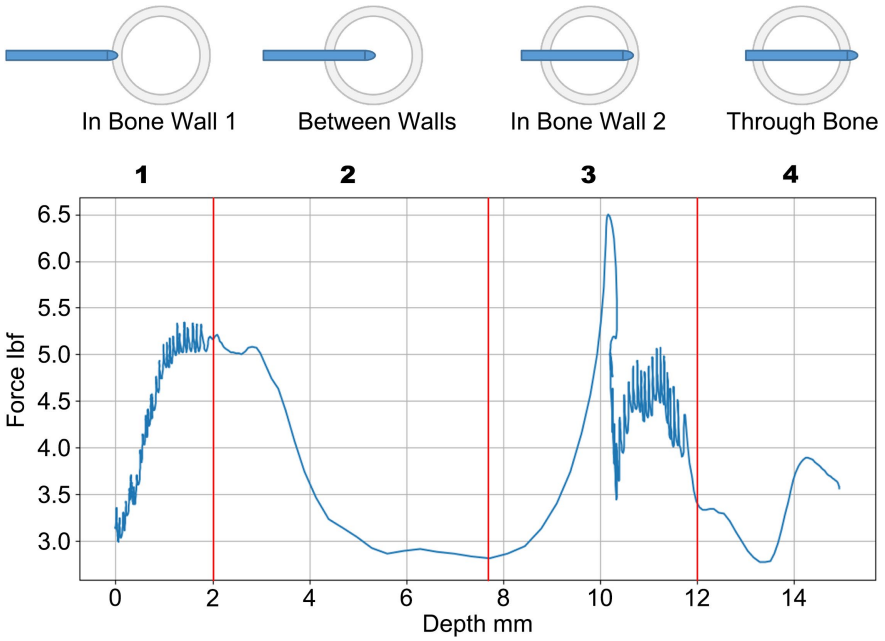

Figure 1. (Top) Simplified illustration of drilling through a bone. (Bottom) A window of data from an ideal sequence of data points showing the drill in use.

In the ideal case, the procedure progresses monotonically with respect to depth, and the transitions between classes can be clearly visualized using the relationship

between depth and force. The applied force should be high when drilling on bone and low when drilling elsewhere. For the change in depth over time (velocity) the opposite is true; velocity should be low when drilling on bone and high when drilling elsewhere. However, noise introduced by the surgeon and/or depth sensor can make sections of the procedure appear to run in reverse, as the depth may occasionally decrease. Also, surgeon-induced oscillations and noise from the force sensor can obscure the transitions between regions.

Provided accurate classification of the data instances, the surgeon will know exactly where the drill is in the bone and the drill will automatically stop when it has pierced the second bone wall (transitioning from region 3 to region 4). But, there are two situations in which incorrect classification may cause undesirable outcomes: if the drill is stopped while in the bone (classifying regions 1, 2, or 3 as region 4), or if the drill is allowed to plunge into the soft tissue (classifying region 4 as either region 1, 2, or 3).

3. Related Work

In the past decade, there have been incredible strides in using DL techniques for time series (TS) forecasting [11] [12], computer vision [13]-[15], and natural language processing (NLP) [16]-[18], but there has been little development of novel deep neural networks (DNN) for TSC [19]. Unlike their counterparts, DL techniques for TSC have only recently begun to provide comparable performance to other state-of-the-art methods. The state-of-the-art TSC methods that DL is commonly compared to are kernel methods with distance functions.

Two causes for the recent improvement of DL techniques for TSC are increased data availability and improved computer vision models. DNNs flourish in data-rich settings, but kernel methods with distance functions struggle to interpret large datasets efficiently. DL computer vision models have improved significantly in the last decade, and some state-of-the-art DNN models for TSC have been adapted from these computer vision models.

It is common to use transforms such as Gramian angular fields or Markov transition fields to represent a TS as an image and then use state of the art image models for classification [20]. It has also become common to adapt state of the art image processing architectures (classification or segmentation) to TSC [21] such as the residual and the inception architectures.

The residual connection, the foundation of residual neural networks (ResNets), was successfully used in deep neural networks by He *et al.* [22] to train deep image classification networks while avoiding the issue of vanishing gradients. The ResNet architecture was later successfully adapted for TSC by Wang *et al.* [23] to address vanishing gradients in TSC models. The ResNet implemented by Wang *et al.* consists of three residual blocks followed by a global average pooling layer and a Softmax output layer. The residual blocks in their work consist of three 1D convolution layers, each followed by a batch normalization layer and a rectified linear unit (ReLU) activation layer. This architecture places the batch normalization

layer before the activation layer which has been experimentally proven to be incorrect [24]. This architecture mimics their fully convolutional network (FCN) to allow an accurate comparison of the two models, but it is not designed to optimize performance. Thus, this work is only a demonstration of a working ResNet for TSC. They used the Adam optimizer and categorical cross entropy (CCE) loss to train their ResNet model on the UCR time series archive [25] and found that ResNet could compete with the performance of the state-of-the-art kernel methods.

The DNN inception architecture was introduced by Szegedy *et al.* [26] to increase model representational power for image segmentation and was successfully adapted for TSC by Fawaz *et al.* [27] in their Inception Time network. The Inception Time network consists of two residual blocks, each spanning three inception modules, a global average pooling layer after the inception portion of the network, and a linear layer classifier after the pooling layer. Under the network architecture of Inception Time, an inception module consists of a bottleneck layer followed by three parallel 1D convolution layers and an average pooling layer (in parallel with the convolution layers), with all four parallel layers concatenated into one output. The bottleneck layer either reduces the dimension of a multivariate input, through multi-channel 1D convolution layers, or maintains the dimension of a univariate input, through an identity layer. The three convolution layers process the output of the bottleneck layer with various kernel sizes to learn information present at different sample rates. They used the Adam optimizer to train their Inception Time network on the UCR time series archive [25] and found that their network's performance is comparable to the state-of-the-art kernel methods as well as the ResNet network developed by Wang *et al.* [23].

Research on DNN for TSC continues to progress without addressing IDIC, as evidenced by the absence of publications on the subject. Instead, research on DNN for TSC almost exclusively focuses on whole sequence classification. In this research, the use of state-of-the-art DNN for whole sequence TSC on the task of IDIC is explored. This is done by comparing the performance of state-of-the-art architectures and baseline networks on the task of localizing a drill while it traverses a bone.

4. Data Preprocessing

The curated data used in this research was gathered by researchers using a drill that utilizes depth (distance) and forces sensors to track the drill bit's position while boring holes in cadavers and bone-like substances. With input from domain experts, three change points are identified in the data. A typical sequence is shown in **Figure 2**, where the force applied to the patient is plotted with respect to the distance traversed in the bone, and vertical red lines identify the three change points. It can be seen that the four regions or classes include data in the ranges [0, 2] mm, [2, 8] mm, [8, 12] mm, and [12, 15] mm. The two peaks represent the drill bit penetrating the first and second walls of the bone, and the parabolic region with a flat bottom depicts the drill bit's traversal between the bone walls.

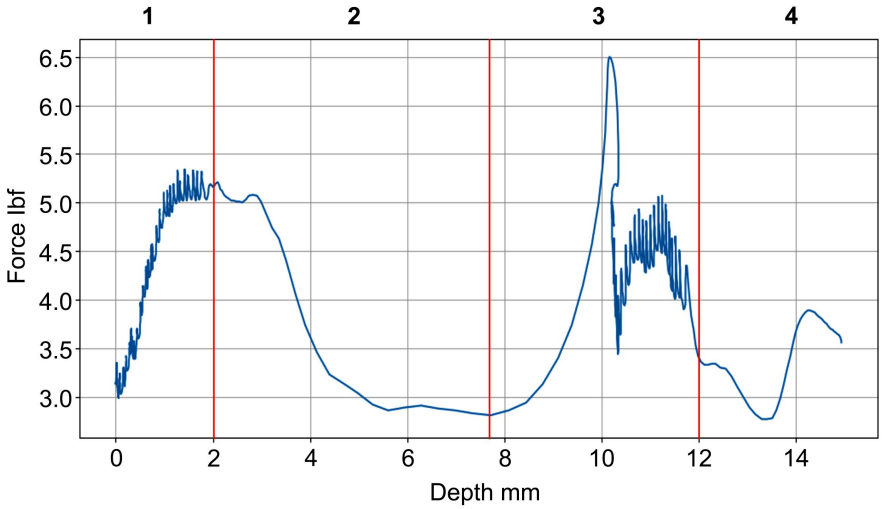

Figure 2. A window of data from an ideal sequence of data points showing the drill in use.

4.1. Preliminary Data Analysis

The dataset consists of 2312 data sequences, resulting in a total of 39,222,088 individual data instances. In Table 1, the maximum value, minimum value, global mean (μ), global standard deviation (σ), average sequence mean ($\hat{\mu}$), and average sequence standard deviation ($\hat{\sigma}$) of the two input features force and distance are presented. The global μ and σ are computed over all data samples. In contrast, the average sequence $\hat{\mu}$ and $\hat{\sigma}$ are computed by finding the μ and σ of each sequence (a drill operation) and then averaging over the number of sequences processed. In this data, sequence length is not constrained because it is dependent on various factors such as the thickness of the bone, the surgeon, etc. Hence, longer sequences will have more influence on the distribution than shorter ones. To address the sequence length variability, the average sequence statistics are computed irrespective of sequence length.

Table 1. Analysis of raw data.

Feature	Max	Min	μ	σ	$\hat{\mu}$	$\hat{\sigma}$
Depth mm	86.470	−76.260	5.619	14.043	6.344	9.807
Force lbf	21.990	−4.630	3.017	2.400	2.762	1.075

The device's depth sensor is configured to operate in the range [0, 64] mm, and the device must follow the physical limitation of positive depth during the operation. Similarly, the force measurements should be positive with the drill being pushed through the bone. However, the force may be negative due to calibration issues of the sensor, rebounding of the drill bit from a collision with bone or trabeculae (floating mass inside the bone), and the drill being pulled back at the end of the procedure. Therefore, the minimum and maximum values of the depth and force from Table 1 imply that the data must be preprocessed.

4.2. Data Preprocessing

The preprocessing steps performed restrict the data to the operation range, reduce noise, and normalize the features of ML. The four preprocessing steps include truncation, positive monotonic restriction, pruning, and normalization.

Truncation (the process of removing data from a series) was used to prune data recorded after the end of the procedure by limiting each sequence to the range $[t_0 : t_{\max}]$. Where t_0 represents the time of the first zero depth measurement and $t_{\max}$ represents the time when the drill achieved its maximum depth for that sequence.

To reduce noise from the surgeon and the sensors, the recorded depth measurements were constrained to be positive monotonic. This constraint removes measurements where the drill is stationary or moving backward and simplifies the data by ensuring the sequences progress linearly between classes. The force measurements were also removed at the corresponding times to maintain the temporal alignment of the data. The validity of the assumption that the procedure may be restricted to positive velocity was verified by the domain experts.

After truncation and monotonic restriction, the sequences that could not be labeled with high confidence were pruned to clean the data further. This included sequences that deviated significantly from the examples provided by the domain experts and series in which too much noise was present to label the data accurately. Label confidence was decided based on how much the series diverged from the examples labeled by the domain expert.

Feature normalization was performed by constraining the data to the range [0, 1], using the maximum sensor value for Depth mm (64 mm) and the largest measured value for force (22 lbf). The common method of z-normalization was not used because performing a zero mean shift on the data would violate the physical constraint of positive depth. After preprocessing, the data consists of 1,375,999 samples among 1341 series.

Data partitioning into sets was conditioned on label confidence. The labels with the highest confidence were provided by a domain expert, and this data was used for the validation and test datasets. The labeling for the training set was performed by studying the validation and test datasets. The data distribution under this partitioning scheme was 87.33% (1013 sequences) training, 7.16% (199 sequences) validation, and 5.51% (129 sequences) test. As was introduced in section 1, an inherent imbalance is present in this data. After labeling, the portion of the total data instances represented by each class is 28.54% (class 1), 12.15% (class 2), 41.04% (class 3), and 18.26% (class 4). The statistics per class are shown in Table 2 and the class-colored histograms are shown in Figure 3.

The statistics in Table 2 show the labeled data is much closer to the valid range for the depth sensor, and the magnitude of the negative force data has been reduced. For the depth data, μ and σ are larger than $\hat{\mu}$ and $\hat{\sigma}$ for all classes. Because the average sequence statistics are both smaller than the global statistics, there exist long sequences that have higher μ and σ than the average

sequence. The reduction in the negative force measurements could be due to the removal of the data from drill extraction, which occurs after the maximum depth has been reached.

Table 2. Labeled data statistics by class.

Feature - Class	Max	Min	μ	σ	$\hat{\mu}$	$\hat{\sigma}$
Depth mm - 1	26.360	−0.670	3.687	3.022	2.567	1.254
Depth mm - 2	42.720	0.010	13.015	7.004	9.771	2.983
Depth mm - 3	57.550	0.610	22.413	9.157	17.118	1.575
Depth mm - 4	63.530	0.960	28.237	11.138	25.359	2.562
Force lbf - 1	21.990	−0.990	3.241	3.010	3.437	0.569
Force lbf - 2	21.990	−0.950	3.193	2.620	3.416	0.464
Force lbf - 3	21.990	−0.580	3.627	2.650	3.816	0.362
Force lbf - 4	21.990	−1.630	3.290	2.651	3.353	0.512

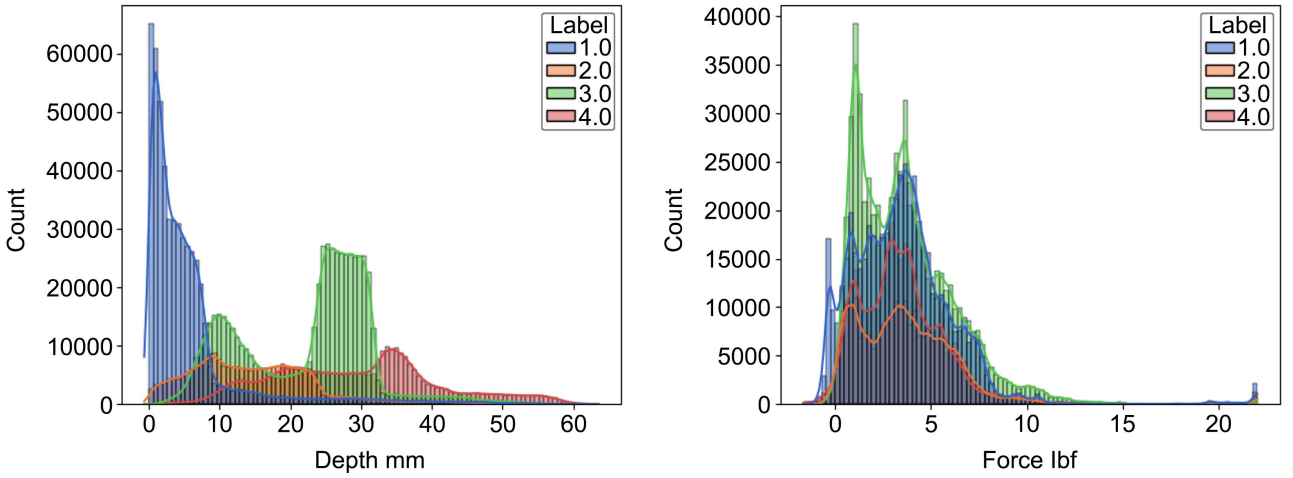

Figure 3. Labeled data subset histograms.

Theoretically, the force values should have a high magnitude when the drill is in contact with the bone and a low magnitude when drilling elsewhere; however, this relationship does not hold well in this dataset. This point is evident in the histogram in **Figure 3** and substantiated by the approximately equal μ and small σ for each class of the force data in **Table 2**.

In **Figure 3**, the histograms show significant overlap between classes in both depth and force data. Although the force data does not provide any differentiation between classes, the force has been included, as suggested by the domain experts, to identify perturbations. A perturbation is a collision between the drill and trabeculae (floating materials between the walls of a bone), which causes the force to be near zero or negative and the depth to decrease or stagnate.

5. ML Architectures and Training

The three previously established requirements for an automated instrument localization algorithm are reiterated below:

1) Temporally/spatially aware
2) Able to process individual data instances
3) Capable of generalizing to new data

The first requirement was addressed through the architecture selection: CNN and RNN. In this section, the other two requirements are addressed by investigating the data loading process and examining network inference performance on the test set.

ML for IDIC is not currently widely researched; thus, this work explores the use of various network architectures to solve the problem of IDIC. This section first provides the model architectures and hyperparameters, followed by the training and evaluation procedures, and finally, a discussion of the inference results.

5.1. Architectures

The models constructed in this research were broken into two stages: a temporal transformation stage and a classification stage. First, in the temporal transformation stage, the input time series is transformed into a latent space by convolution or recurrent layers. Then, in the classification stage, the latent vectors are classified by linear layers. The layers for these two stages were grouped into repeatable blocks to simplify the architecture. The convolution, recurrent, and linear blocks are shown in **Figure 4**. The baseline network architectures are presented in **Table 3**. The input width for these models is 32 data instances and the output is a softmax layer with 4 neurons.

The regularization techniques, dropout [28] and batch normalization [29], were included to prevent overfitting and to increase generalization. A dropout percent of (25%) was determined adequate heuristically for RNN 1 and was then maintained for all networks.

The advanced network architectures include Residual networks (ResNets) and Inception networks. These networks use the same block structure as was discussed previously and have softmax output layers with 4 neurons. The advanced network architectures are shown in **Table 4**.

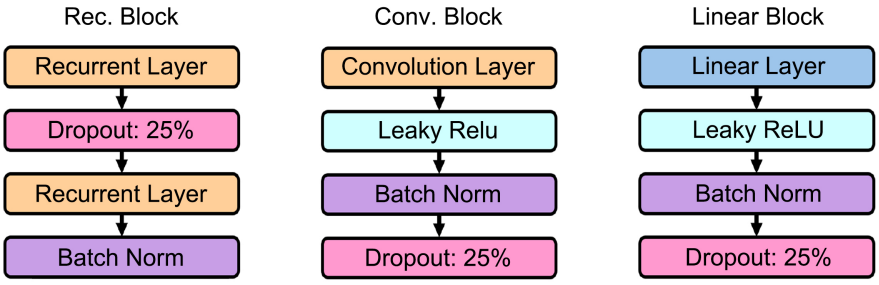

Figure 4. A diagram of the layer configuration used in a recurrent (rec.) block, a convolution (conv.) block, and a linear block.

Table 3. Baseline model architectures.

Name	RNN 1	RNN 2
Layer type	LSTM	LSTM
Input width	32	32
# Rec. blocks	1	1
Rec. layer width per block	256	64
# Linear blocks	3	3
Linear layer width per block	32, 16, 8	32, 16, 8
Output width	4	4
Name	RNN 3	RNN 4
Layer type	GRU	GRU
Input width	32	32
# Rec. blocks	1	1
Rec. layer width per block	256	128
# Linear blocks	3	3
Linear layer width per block	32, 16, 8	32, 16, 8
Output width	4	4

Table 4. Advanced model architectures.

Name	ResNet 1	ResNet 2
Layer type	Conv. 1D	Conv. 1D
Input width	32	32
# Residual blocks	2	8
Conv. layer width per block	64	64
# Linear blocks	1	0
Linear layer width per block	16	N/A
Output width	4	4
Name	Inception 1	Inception 2
Layer type	Conv. 1D	Conv. 1D
Input width	32	32
# Inception blocks	8	8
Conv. layer width per block	64	64
# Linear blocks	1	0
Linear layer width per block	16	N/A
Output width	4	4

The model architecture space was explored in a binary search fashion. A non-trivial model that underfits the training data was found, and a model that overfits the training data was found. Finally, the space between these two models was explored. A total of 10 models were found for this research, two models using each temporally aware layer type and each advanced architecture type. The advanced model architectures are shown in Table 4, which follows the same block notation as in Table 3.

5.2. Training

For training, the data was windowed to 32 data instances and grouped into batches of 128 windows by a custom data-loader. To maintain temporal relevance, data within the windows was not shuffled, and windows were not allowed to bridge sequences. The maximum number of epochs was set to 2048 with early stopping patience of 50 epochs and checkpoint saving, both conditioned on improvement in the validation loss. For all models, the learning rate was held constant at 0.0001, and optimization was performed using the Adam optimizer.

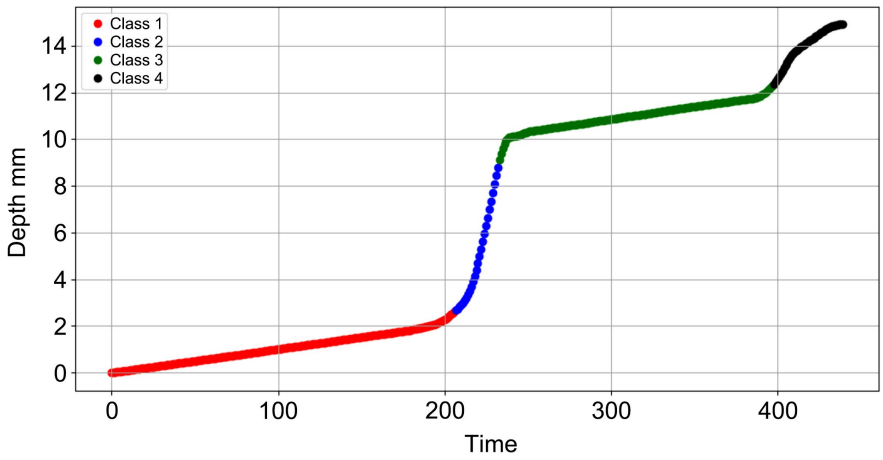

Figure 5. Classification using depth vs time.

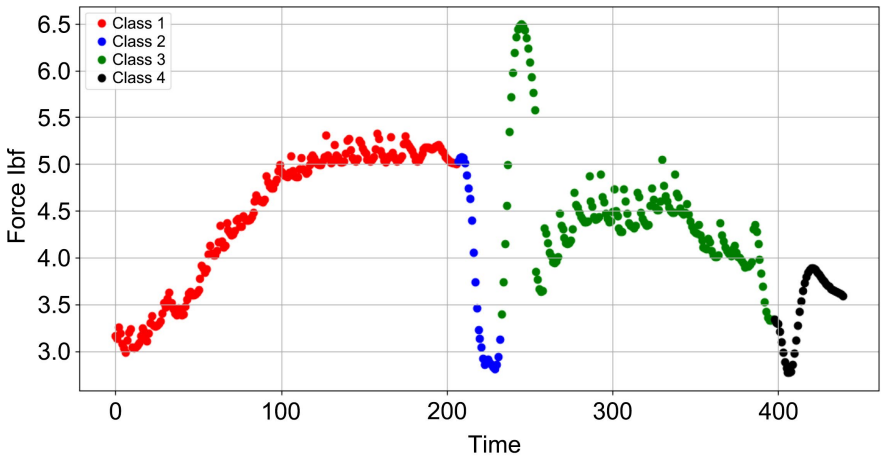

Figure 6. Classification using force vs time.

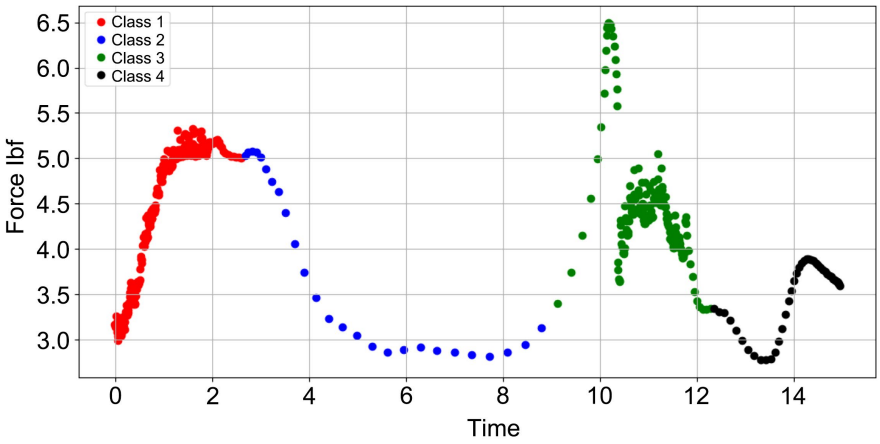

Figure 7. Classification using force vs depth.

Using the device's depth and force sensors as input features, the models were trained on three datasets: two univariate and one bivariate. The two univariate datasets were constructed from each input feature individually. The bivariate dataset includes both input features. Example plots showing data from the same labeled sequence, but from each dataset, are shown in Figure 5 through Figure 7.

6. Results

The performance of all models considering the accuracy of each class for the test data set is shown in Table 5. Examining the prediction accuracy of the inference models on each class, the following salient performance characteristics can be identified:

- For all inference models, the prediction accuracy is significantly higher on the univariate depth dataset than on the univariate force dataset. It can be inferred that the models have learned to classify the univariate depth data but have not learned to classify the univariate force data.
- Examining the univariate force sequence shown in Figure 6, there are multiple instances where the magnitude of the force is the same. This may be the cause for the low inference accuracy on the univariate force dataset.
- The univariate depth dataset exhibits well-defined boundaries for each class, which can be seen in Figure 5, whereas the univariate force dataset does not exhibit similar boundaries for each class.
- For all inference models, the performance with the bivariate dataset is comparable to the performance with the univariate depth dataset. Since the same or higher accuracy is achieved with the univariate depth data, the models may have learned to rely only on the depth patterns for inference.
- Considering the individual prediction accuracy of each class, RNN 3 has achieved the best performance of the baseline models, and ResNet 2 and Inception 2 have achieved the best performance among the advanced models.
- The accuracy of models RNN 3, ResNet 2, and Inception 2 is low in classes 2 and 4 compared to the accuracy in classes 1 and 3. The data instances of class

2 are from the region between bone walls, while class 4 data instances are from the region beyond the second bone wall where the drill has plunged into the surrounding tissue. In both regions, the number of data instances is significantly less than that in the regions representing classes 1 and 3. The percent of data represented by each class is 28.24% (class 1), 12.26% (class 2), 41.95% (class 3), and 17.55% (class 4). This imbalance in the dataset is caused by the constant sampling rate of the sensors on the drill accompanied by a faster traversal speed in class 2 and 4 regions. Due to this imbalance in the training dataset, the training is biased towards learning classes 1 and 3.

- All of the best performing models have low class 2 accuracy with an average accuracy of 34%. This can be due to a low number of class 2 data instances in the training dataset.
- The ResNet 2 model has the best overall accuracy according to class accuracy performance.

Table 5. Composite inference accuracy for all models trained with original data.

Model	Feature set	Class 1 (%)	Class 2 (%)	Class 3 (%)	Class 4 (%)
*RNN 1 (LSTM)	Depth	96.6	35.2	73.5	39.0
	Force	14.9	12.2	63.8	7.6
	Bivariate	94.6	34.9	72.4	24.4
*RNN 2 (LSTM)	Depth	97.0	33.9	75.5	39.0
	Force	14.2	9.5	71.2	8.8
	Bivariate	97.4	34.4	71.7	21.3
*RNN 3 (GRU)	Depth	96.1	33.6	76.9	57.4
	Force	22.7	7.4	65.7	5.6
	Bivariate	98.2	29.5	70.8	22.6
*RNN 4 (GRU)	Depth	95.4	32.9	77.3	56.3
	Force	13.6	10.3	76.3	4.9
	Bivariate	95.6	34.1	74.8	27.3
*ResNet 1	Depth	96.0	33.2	74.2	65.8
	Force	20.0	11.6	66.6	6.0
	Bivariate	96.7	34.5	60.2	18.6
*ResNet 2	Depth	93.5	34.8	75.3	68.3
	Force	15.4	6.1	68.6	5.7
	Bivariate	96.4	35.0	62.5	15.3

Continued

	Depth	96.2	35.7	62.5	59.0
*Inception 1	Force	28.5	21.5	51.9	10.9
	Bivariate	97.3	23.2	50.6	29.2
	Depth	95.3	34.5	80.9	56.2
*Inception 2	Force	25.2	20.5	65.9	5.8
	Bivariate	95.6	27.1	52.9	25.6

Since the dataset consists of strongly imbalanced classes, the accuracy-based performance of the models can be misleading. From surveying the literature for metrics specific to IDIC, it was determined that the precision and recall performance metrics would adequately describe model performance. Therefore, the performance is analyzed further for only RNN 3, ResNet 2, and Inception 2 models using the precision and recall performance metrics.

- Precision metric (probability) is the ratio of correct positive predictions to total positive predictions (sum of true and false positives). The precision for the positive class does not imply anything for the negative classes.
- Recall metric (probability) is the ratio of correct positive predictions to the number of positive labels (all samples that should be classified as positive).

The precision and recall metrics of the three models are presented in Table 6, computed from the confusion matrices in Table 7. Examining the confusion matrices of all three models, it can be inferred that all three models perform poorly in classifying class 2 data instances, with a 65% false positive rate. This inference is further validated by the very low (0.34 probability) precision and recall metrics of all the models. The recall metric for class 1 is significantly higher (0.93) compared to the precision metric (0.70), indicating that all of the models are correctly identifying data instances belonging to class 1 as class 1, *i.e.*, with respect to class 1 classification, the models are able to generalize. Similar performance with class 3 can be observed since both the precision (0.80) and recall metrics (0.76) are high and approximately the same. However, in the case of class 4, the recall metric is low for all models, indicating that all models are classifying a significant number of class 4 data instances as other classes. This uneven performance can be attributed to the strongly imbalanced classes in the training dataset biasing the network models to learn the characteristics of classes 1 and 3 only.

Furthermore, studying the training loss shown in Figure 8, it can be inferred that RNN 3 is performing better compared to ResNet 2 and Inception 2 models. The ResNet 2 and Inception 2 models exhibit overfitting as the validation loss does not decrease with the corresponding decrease in the training loss. In contrast, the RNN 3 validation loss decreases with the decrease in the training loss.

Table 6. Performance metrics precision and recall for models RNN3, ResNet 2, and Inception 2.

Model	Metric	Class 1	Class 2	Class 3	Class 4
*RNN 3 (GRU)	Precision	0.6969	0.3361	0.7982	0.9133
	Recall	0.9611	0.3366	0.7690	0.5741
*ResNet 2	Precision	0.6960	0.3417	0.8278	0.9198
	Recall	0.9344	0.3482	0.7538	0.6834
*Inception 2	Precision	0.7280	0.3558	0.7716	0.9437
	Recall	0.9536	0.3450	0.8092	0.5628

Table 7. Confusion matrices for models RNN 3 (top), ResNet 2 (middle), and Inception 2 (bottom).

RNN 3 (GRU)				
Pred \ Label	1	2	3	4
1	24,516	7619	2843	202
2	941	4144	3994	3252
3	45	309	26,067	6235
4	6	241	992	13,058
ACC	0.9611	0.3366	0.7690	0.5741
ResNet 2				
Pred \ Label	1	2	3	4
1	23,853	7338	2623	339
2	1606	4287	4690	1962
3	36	377	25,551	4901
4	13	311	1032	15,545
ACC	0.9351	0.3482	0.7538	0.6834
Inception 2				
Pred \ Label	1	2	3	4
1	24,324	7529	1519	39
2	1016	4248	4273	2403
3	168	449	27427	7502
4	0	87	677	12,803
ACC	0.9536	0.3450	0.8092	0.5628

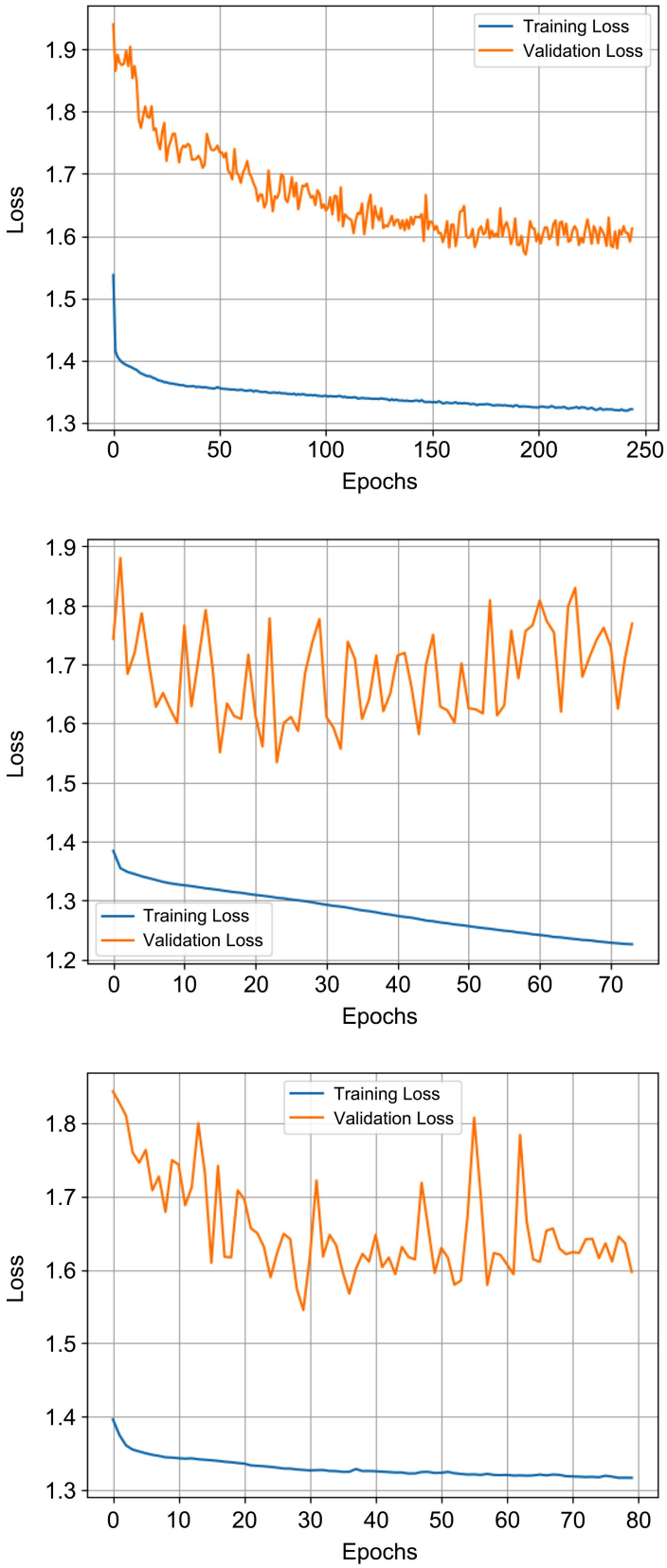

Figure 8. Plots of training and validation loss for models RNN 3 (left), ResNet 2 (right), and Inception 2 (bottom), trained with the univariate depth dataset.

7. Conclusions and Future Work

The plunge depth and hardware placement in existing ORIF methods to set displaced distal radius fractures can be improved by providing the surgeon with an accurate locality of the drill during the operation. In this work, automatic instrument localization leveraging DL models for time series IDIC was evaluated on a dataset from a drill that has a force sensor and a depth sensor. After data preprocessing, CNN and RNN architectures, including state-of-the-art residual and inception architectures, were evaluated. The significant performance results of the models are presented below:

- The data from the depth sensor has well-defined boundaries for each class, and all of the DL models were able to learn when training with the univariate depth dataset.
- The data from the force sensor does not have well-defined boundaries for each class, and therefore none of the models learned to classify the data using the univariate force dataset.
- The best-performing baseline model was RNN 3, and the best-performing advanced models were ResNet 2 and Inception 2. However, ResNet 2 and Inception 2 both exhibited overfit during training.
- All models performed significantly better at classifying data from drilling on the bone walls (classes 1 and 3) than at classifying data from drilling elsewhere (classes 2 and 4), due to the imbalance in this data.

It may be possible to improve classification performance by balancing the dataset prior to training. Since it is not possible to change the sampling rate of the sensors of the drill or to require a surgeon to traverse at a lower speed in the regions corresponding to classes 2 and 4, both removing data instances from the overrepresented classes and generating synthetic data instances for the underrepresented classes should be investigated.

Acknowledgement

We express our sincere thanks to the domain experts for providing help with labeling the datasets and insights about medical procedures.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Maccagnano, G., Noia, G., Vicenti, G., Baglioni, M., Masciale, M.R., Cassano, G.D., *et al.* (2021) Volar Locking Plate versus External Fixation in Distal Radius Fractures: A Meta-Analysis. *Orthopedic Reviews*, **13**, Article No. 9147. https://doi.org/10.4081/or.2021.9147

[2] del Piñal, F., Jupiter, J.B., Rozental, T.D., Arora, R., Nakamura, T. and Bain, G.I. (2021) Distal Radius Fractures. *Journal of Hand Surgery* (*European Volume*), **47**, 12-23. https://doi.org/10.1177/17531934211028711

[3] Stoops, T.K., Santoni, B.G., Clark, N.M., Bauer, A.A., Shoji, C. and Schwartz-Fernandes, F. (2016) Sensitivity and Specificity of Skyline and Carpal Shoot-Through Fluoroscopic Views of Volar Plate Fixation of the Distal Radius: A Cadaveric Investigation of Dorsal Cortex Screw Penetration. *HAND*, **12**, 551-556. https://doi.org/10.1177/1558944716677336

[4] DeGeorge, B.R., Brogan, D.M., Becker, H.A. and Shin, A.Y. (2020) Incidence of Complications Following Volar Locking Plate Fixation of Distal Radius Fractures: An Analysis of 647 Cases. *Plastic & Reconstructive Surgery*, **145**, 969-976. https://doi.org/10.1097/prs.0000000000006636

[5] Choi, J.H., Lee, Y.S., Hwang, K., Jo, Y., Shin, H.S., Kim, J., *et al.* (2023) Usefulness of a Drill Stopper to Prevent Iatrogenic Soft Tissue Injury in Orthopedic Surgery. *Heliyon*, **9**, e20772. https://doi.org/10.1016/j.heliyon.2023.e20772

[6] Maschke, S.D., Evans, P.J., Schub, D., Drake, R. and Lawton, J.N. (2007) Radiographic Evaluation of Dorsal Screw Penetration after Volar Fixed-Angle Plating of the Distal Radius: A Cadaveric Study. *HAND*, **2**, 144-150. https://doi.org/10.1007/s11552-007-9038-2

[7] Perry, J., Collins, A. and Gilmer, B. (2021) Dual Motor Orthopaedic Drill Reduces Plunge and Provides Data for Safe and Accurate Screw Placement during Clavicle Fracture Fixation. *Journal of Orthopaedic Experience & Innovation*, **2**, 1-8. https://doi.org/10.60118/001c.18357

[8] Hubly Surgical: Hubly Drill. https://hublysurgical.com/

[9] Mavrogenis, A.F., Savvidou, O.D., Mimidis, G., Papanastasiou, J., Koulalis, D., Demertzis, N., *et al.* (2013) Computer-Assisted Navigation in Orthopedic Surgery. *Orthopedics*, **36**, 631-642. https://doi.org/10.3928/01477447-20130724-10

[10] Christen, B., Tanner, L., Ettinger, M., Bonnin, M.P., Koch, P.P. and Calliess, T. (2022) Comparative Cost Analysis of Four Different Computer-Assisted Technologies to Implant a Total Knee Arthroplasty over Conventional Instrumentation. *Journal of Personalized Medicine*, **12**, Article No. 184. https://doi.org/10.3390/jpm12020184

[11] Mahalakshmi, G., Sridevi, S. and Rajaram, S. (2016) A Survey on Forecasting of Time Series Data. 2016 *International Conference on Computing Technologies and Intelligent Data Engineering* (*ICCTIDE*'16), Kovilpatti, 7-9 January 2016, 1-8. https://doi.org/10.1109/icctide.2016.7725358

[12] Murray, C., Chaurasia, P., Hollywood, L. and Coyle, D. (2022) A Comparative Analysis of State-of-the-Art Time Series Forecasting Algorithms. 2022 *International Conference on Computational Science and Computational Intelligence* (*CSCI*), Las Vegas, 14-16 December 2022, 89-95. https://doi.org/10.1109/csci58124.2022.00021

[13] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., *et al.* (2015) ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, **115**, 211-252. https://doi.org/10.1007/s11263-015-0816-y

[14] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., *et al.* (2023). Segment Anything. 2023 *IEEE/CVF International Conference on Computer Vision* (*ICCV*), Paris, 1-6 October 2023, 3992-4003. https://doi.org/10.1109/iccv51070.2023.00371

[15] Wang, C., Bochkovskiy, A. and Liao, H.M. (2023) YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. 2023 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Vancouver, 17-24 June 2023, 7464-7475. https://doi.org/10.1109/cvpr52729.2023.00721

[16] Cambria, E. and White, B. (2014) Jumping NLP Curves: A Review of Natural Language Processing Research [Review Article]. *IEEE Computational Intelligence Magazine*,

9, 48-57. https://doi.org/10.1109/mci.2014.2307227

[17] OpenAI Inc. (2024) Gpt-4 Technical Report.

[18] Manyika, J. and Hsiao, S. (2023) An Overview of Bard: An Early Experiment with Generative AI.

[19] Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L. and Muller, P.-A. (2019) Deep Learning for Time Series Classification: A Review. Data Mining and Knowledge Discovery.

[20] Faozi, J. (2022) Time Series Classification: A Review of Algorithms and Implementations. In: *Machine Learning* (*Emerging Trends and Applications*), Proud Pen, 1-34.

[21] Ruiz, A.P., Flynn, M., Large, J., Middlehurst, M. and Bagnall, A. (2020) The Great Multivariate Time Series Classification Bake off: A Review and Experimental Evaluation of Recent Algorithmic Advances. *Data Mining and Knowledge Discovery*, **35**, 401-449. https://doi.org/10.1007/s10618-020-00727-3

[22] He, K., Zhang, X., Ren, S. and Sun, J. (2016) Deep Residual Learning for Image Recognition. 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, 27-30 June 2016, 770-778. https://doi.org/10.1109/cvpr.2016.90

[23] Wang, Z., Yan, W. and Oates, T. (2017) Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline. 2017 *International Joint Conference on Neural Networks* (*IJCNN*), Anchorage, 14-19 May 2017, 1578-1585. https://doi.org/10.1109/ijcnn.2017.7966039

[24] Rosebrock, A. (2019) Deep Learning for Computer Vision. PyImageSearch.

[25] Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A. and Batista, G. (2015) The UCR Time Series Classification Archive. https://www.cs.ucr.edu/eamonn/time_series_data/

[26] Szegedy, C., Liu, W., Jia, Y.Q., Sermanet, P., Reed, S., Anguelov, D., *et al.* (2015) Going Deeper with Convolutions. 2015 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Boston, 7-12 June 2015, 1-9. https://doi.org/10.1109/cvpr.2015.7298594

[27] Fawaz, H.I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D.F., Weber, J., Webb, G.I., Idoumghar, L., Muller, P.-A. and Petitjean, F. (2020) InceptionTime: Finding AlexNet for Time Series Classification. Data Mining and Knowledge Discovery.

[28] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014) Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, **15**, 1929-1958.

[29] Ioffe, S. and Szegedy, C. (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Proceedings of the* 32*nd International Conference on Machine Learning*, Lille, **37**, 448-456.