Character Recognition on Time Series Data collected from Smartphone Sensors

Deep Raval\textsuperscript{1}, Jaymin Suhagiya\textsuperscript{1} and Sukriti Macker\textsuperscript{2}
\textsuperscript{1}Department of Information and Communication Technology, Adani Institute of Infrastructure Engineering, Gujarat, India
\textsuperscript{2}Department of Computer Science Engineering, Bhagwan Mahaveer College of Engineering and Management, Haryana, India
E-mail: deepraval.ict17@gmail.com

Abstract. In the modern era, smartphones have become part and parcel of life. The exponential increase in the number of smartphone users has given rise to a copious amount of data. Although the data produced by smartphones are of various types, this paper mainly focuses on the usage of sensory data produced by smartphones. This paper explores the field of recognizing characters with the aid of time-series sensor data. The primary focus of the research is to utilize recurrent neural networks to predict the digits $0 \to 9$ and characters $A \to Z$. The objective achieved was by the help of sensor data that included the readings of Accelerometer, Magnetometer, Gyroscope, and Linear Accelerometer sensors providing information with respect to three-axis $x$, $y$, and $z$, having an interval of 0.01 seconds between two corresponding values. We succeeded in achieving the accuracy of 93.60\% on the training data and 89.51\% on the testing data.

1. Introduction
Character recognition is an active research field that has been persistently on the radar for various development and research projects. Earlier, character recognition was achieved by the use of extracting features, and the key focus was on segmented characters. The Modified National Institute of Standards and Technology (MNIST) dataset consists of digits written by hand [1]. It is a profound example of an image-based character recognition dataset widely used by professionals and researchers. The contribution of neural networks has been of immense value providing a test error rate (%) as low as 0.35 [2]. Furthermore, Convolutional Neural Networks (CNN) overwhelmingly manage multi-dimensional data [3], providing a phenomenal test error rate (%) of 0.23 [4]. To conclude, the possibilities provided by the neural networks in the domain of character recognition are limitless. Therefore, this paper further explores the possibility of determining characters based on time-series data acquired from smartphone sensors.

The paper by Christoph Amma et al. [5] proposed a wearable computing device that consists of sensors like the accelerometer and gyroscope. Their predictions used Hidden Markov Models (HMM) [6], which, in turn, achieved an accuracy of 81.9\%. The paper by Muhammad Arsalan et al. [7] used three $60GHz$ radars to obtain the data. Their model was based on Conv-LSTM [8] achieved an accuracy of 98.33\%. However, its applicability was limited as they only considered the digits $1 \to 5$ and the characters $A \to J$. 
Various parameters come under consideration like movements, which are linear, orientation, and others while examining the character’s production in the air using a smartphone. The utilization of smartphone sensors to acquire data while writing a character in the air is useful because of the raw sensor data it provides. This data demands less pre-processing as compared to multi-dimensional data like images. The ever-growing usage of smartphones and its wide availability makes the research efficiently deployable. Moreover, sensor data’s immediate employment provides an advantage over multi-dimensional data, and it is because writing on a screen or a paper requires heavy pre-processing before its utilization.

This paper studies the applicability of four sensors that are Magnetometer, Accelerometer, Linear Accelerometer, and Gyroscope. The magnetometer measures (in $\mu$T) the direction and strength of magnetism or changes in the magnetic field in 3 dimensions. It provides a 3D vector that steers towards the most powerful magnetic field, or else it points towards the magnetic field of the earth if it does not detect the presence of a strong magnetic field around it. Accelerometer measures changes in acceleration ($m/s^2$), which results in understanding an object’s inclination in 3 dimensions. Its value is affected by acceleration caused by gravity. The Linear Accelerometer obtains the values through the accelerometer and either magnetometer or gyroscope [9] and is measured in ($m/s^2$). The Gyroscope senses and measures the angular velocity. It senses the orientation and rotational motion in 3 dimensions relative to itself. It is expressed in ($rad/s$).

In past studies, recurrent neural networks have provided phenomenal results on the time-series data [10]. Both Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) cells were tested during the experimentation. The GRU and LSTM cell were first introduced in [11] and [12], respectively. These cells have the authority to discard the information or the data which is already stored. The function that they present plays a crucial role that serves our aim to strengthen our neural network to make its own decisions to know when and how the values are to be molded while the character is being drawn. Figure 1 depicts the structure of the LSTM cell, and Figure 2 portrays structure of the GRU cell, where $\sigma$ is the representation of the sigmoid function. Equation 2 represents the $tanh$ function, and Equation 1 represents the sigmoid function. Although GRU and LSTM both exercise sigmoid and $tanh$ as the activation functions that can be modified, we used it as it is. Variations of LSTM like Convolutional [8] and Bidirectional [13] were also tested during the experiments.

\[
sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{1}
\]
\[
tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2}
\]

The paper’s organization is as follows: Section 2 describes the dataset used; Section 3 describes the architecture of proposed models; Section 4 describes the experimental results.
observed and the evaluation of the models. At last, Section 5 presents conclusions and future work.

2. Exploring The Dataset
Due to the lack of a state-of-the-art dataset for the task, the dataset was made from the ground up. This segment explores the steps of Data Acquisition- the process of obtaining the data, Data Analysis- understanding and analyzing the statistical information extracted from the data, Data Preprocessing- the method of cleaning, pre-processing, and enhancing the data that is to be fed into the intended model.

Figure 3: Hand movements of a volunteer while writing the character ‘C’. The blue line shows the trajectory, while the blue dots indicate where the snapshots were taken. This figure also describes how a smartphone is held while writing a character.

Figure 4: The graphical representation of the variations depicted by the value versus timestamp graphs of Accelerometer, Gyroscope, Linear Accelerometer, and Magnetometer while writing the character ‘C’.
2.1. Acquiring the data
It was a slightly challenging job to generalize an unbalanced dataset due to the limited number of sinistral volunteers. The dataset hence created consisted of the data of only six dextral volunteers (including the authors) who were having the age in the range of 17 – 21 years. Age notably is a vital parameter because, considering the movements of the hand, the hand of the elderly is not entirely steady as the hands of a younger individual. Furthermore, unsteady hand movements can lead to increased time of writing a character that can create a noisy dataset that can, in turn, hinder the performance of the model. The instructions provided to write a character in the air was to hold the smartphone in a horizontal manner and proceed in a natural writing method. The specifics of writing a character were not given to any volunteer to contribute to making the dataset diverse. Figure 3 represents the movements of the hand of a volunteer while writing the character ‘C’. Figure 4 portrays the changes in the values of sensor information while writing the character ‘C’. Each sample had a gap of 0.01 seconds between the two timestamps. All the contributions of the volunteers were independent and performed using different smartphones. The dataset has been made open-source, and any individual can use it for research purposes.1

2.2. Statistical Analysis of the dataset
As there was no absolute limit on how much data the volunteers should provide, Figure 6a shows the amount of data gathered in relation to the volunteers. Altogether 5084 examples were gathered due to the presence of corrupted data after the process of elimination, the concluding dataset comprised of 5061 samples. From 5061 samples, 576 (16 samples for every one of 36 classes) samples were randomly selected and saved for the objective of testing. Figure 6b shows the number of samples for every one of 36 classes for the training dataset.

To write, each character’s amount of time varies depending upon the character. Consider writing the character ‘K’ it requires an increased amount of time compared to writing the character ‘O’. Figure 6c portrays the average number of timestamps that represent individual classes in the training dataset. Moreover, Figure 6d depicts similar information for the testing dataset. Further, the value of timestamps may change for writing the same character as well. Every individual may have various styles for composing each character. For example, Figure 5 shows two different approaches to writing the digit ‘1’. Thus, the resulting dataset is not completely balanced and diverse due to the limited number of volunteers.

![Image of the digit 1](https://www.kaggle.com/deepravl2905/charsense-dataset)

Figure 5: The two different styles of writing the digit ‘1’. The digit ‘1’ on the left side has two extra strokes than the digit ‘1’ on the right side.

1 https://www.kaggle.com/deepravl2905/charsense-dataset
2.3. Data Preprocessing

As every sensor had a different scale for the collected values, we applied Min-Max Scaler to scale all of the sensor values in the interval \([-1, 1]\). Equation 3 describes the formula of the Min-Max scaler. Figure 7a and Figure 7b show the comparison between raw values and scaled values.

\[
x' = \left( \frac{x - \text{min}(x)}{\text{max}(x) - \text{min}(x)} \right) \cdot (B - A) + A
\]

Where,
- \(x\) = Original value
- \(x'\) = Scaled value of \(x\)
- \([A, B]\) = Interval of scaled value, \([-1, 1]\) in this case
- \(\text{max}(x)\) = Maximum value of \(x\)
- \(\text{min}(x)\) = Minimum value of \(x\)
3. Proposed Model

Two architectures were selected based on various trials: one for GRU, Bidirectional LSTM, and LSTM based models and one for the Conv-LSTM based model.

Figure 8 shows the first architecture. It is used for Bidirectional LSTM, GRU, and LSTM based models. The respective input and output dimensions are shown in the architecture itself, where an unknown dimension is denoted as ‘?’. The first layer having 144 units, takes the three-dimensional training (or testing) data with tunable batch-size. The output of this layer is passed to the recurrent layer that has 90 units. The fully connected dense layer is the third and fifth layer, having 72 and 36 neurons. The dropout [16] and the recurrent dropout [17] both help in preventing overfitting. Accordingly, the introduction of the dropout layer is done in architecture as the fourth layer. The first and the second layer use both the regular and recurrent dropout. The \textit{elu} function is used as an activation for the third layer. The formula for the same is described in equation 4. The last layer (that is the fifth) layer has 36 units because there are in total of 36 classes. Furthermore, to obtain the probability distribution, the \textit{softmax} function is perfect hence used as an activation function. The formula of the \textit{softmax} function is described in equation 5.

\[
elu(x) = \begin{cases} x, & \text{if } x > 0 \\ e^x - 1, & \text{if } x \leq 0 \end{cases}
\]

\[
\text{softmax}(\vec{z})_i = \frac{e^{z_i}}{\sum_j^k e^{z_j}}, \quad \vec{z} \in \mathbb{R}^k
\]

Figure 7: (a) The raw values of the gyroscope sensor while writing the character ‘C’. (b) The scaled (using MinMax Scaler) values of the gyroscope sensor while writing the character ‘C’.

Figure 8: The proposed architecture for the Bidirectional LSTM, GRU and LSTM based model.
The second architecture is described in Figure 9. On comparing this architecture with the previous one, this architecture essentially has a few actual functioning layers. Another difference is that this architecture only works with the fixed dimension data with the shape \((1, 1, 572, 12)\), as shown in Figure 9. Due to the same reason, training (or testing) instance having timestamps less than 572 needs to be padded, and the instance with more than 572 timestamps must be truncated. The first layer has 64 filters with the \((3, 3)\) kernel. The output of this layer is passed to the flatten layer. As the name suggests, this layer converts multi-dimensional input data to single-dimensional data. This architecture also uses the dropout layer, which is there as the fourth layer. The third layer has 100 units, and the fifth layer having 36 are dense layers; both have the same activation functions as the previous architecture.

Figure 9: The proposed architecture for the Conv-LSTM based model.

4. Experimental Results

Categorical Cross Entropy calculates the difference between two probability distributions. Therefore, it is perfect for classification tasks. Consequently, we used Categorical Cross Entropy as the loss function and Adam [18] as an optimizer. The formula for calculating loss using Categorical Cross Entropy is shown in equation (6). The initialization of all the weights in both architectures was done using the Glorot Uniform initializer [19]. After experimenting with various hyper-parameters, the best model for each category was finalized. Table 1 shows the results of the best model in each category.

\[
Loss = \sum_{i=1}^{\text{vectorsize}} true_i \cdot \log(predicted_i)
\]  

Where,

\(true = \text{target vector}\)

\(predicted = \text{predicted vector by the model}\)

| Cell          | Epochs | Training Accuracy(%) | Testing Accuracy (%) |
|---------------|--------|-----------------------|----------------------|
| LSTM          | 30     | 93.76                 | 88.16                |
| GRU           | 30     | 89.89                 | 88.37                |
| ConvLSTM      | 25     | 98.41                 | 89.77                |
| Bidirectional LSTM | 29 | 93.60                 | 89.51                |

Table 1: The performance of the best models from each category.

Hyper-parameters for the best models from each category: LSTM based model has \(\text{recurrent dropout} = 0.35\), \(\text{dropout} = 0.4\) in the first layer, \(\text{recurrent dropout} = 0.3\), \(\text{dropout} = 0.35\) in
second layer and \[\text{dropout} = 0.3\] in the fourth layer. GRU based model has \[\text{recurrent dropout} = 0.2, \text{dropout} = 0.3\] in the first layer, \[\text{recurrent dropout} = 0.1, \text{dropout} = 0.25\] in second layer and \[\text{dropout} = 0.4\] in the fourth layer. Bidirectional LSTM based model has \[\text{dropout} = 0.25\] in the first layer, \[\text{dropout} = 0.2\] in the second layer and \[\text{dropout} = 0.15\] in the fourth layer. Conv-LSTM based model has \[\text{dropout} = 0.2\] in the first layer and \[\text{dropout} = 0.3\] in the fourth layer. Figure 10 shows the graph of the accuracy versus epoch for all the models. As the graph suggests, training of Conv-LSTM is faster compared to other models thanks to the fixed dimension data that it gets. Due to larger dropout values, LSTM takes longer to achieve similar accuracy compared to others. At last, GRU and Bidirectional LSTM have a similar training curve.

![Graph of accuracy vs epoch](image)

Figure 10: The graph of the accuracy vs epoch for all models.

The confusion matrix of the model based on Bidirectional LSTM is shown in Figure 11. The model frequently has a problem recognizing the digit ‘1’, deducing from the confusion matrix itself. Since numerous characters like ‘I’, ‘J’, ‘7’, and so on contain a vertical line similar to ‘1’, this problem is entirely justifiable. Moreover, different writing styles also contribute to the confusion, for instance, characters like ‘I’ and ‘7’. Considering the dataset was not that large and diverse, the model’s overall performance on testing data is quite good.
5. Conclusions and Future work

This paper explored the character recognition domain on the time-series sensor data. Experimental results concluded that Conv-LSTM based model beats all other models in terms of testing accuracy. Despite the fact that Conv-LSTM based model has the best testing accuracy, it is not reasonable to use it because it expects the data to have a fine-tuned dimension. Apart from Conv-LSTM based model, the Bidirectional LSTM based model can be utilized with the highest testing accuracy among the remaining models. Due to less diversity in the dataset, this achieved testing accuracy can decrease in practical scenarios. For instance, if a left-handed person or an elderly try our model, the results would not be up to the mark because of the same reason.

As the number of volunteers was less, data can be collected from more volunteers to make the dataset diverse and vast. Character recognition is one of the first steps in terms of language processing. The promising results engender a possibility to build a classifier for words rather than the individual characters. As all the proposed models only require data from trivial smartphone sensors, a small wearable gadget can also be made for writing in the 3D space.
References

[1] Deng L 2012 IEEE Signal Processing Magazine 29 141–142
[2] Ciresan D, Meier U, Gambardella L and Schmidhuber J Google Scholar Digital Library Digital Library
[3] Rawat W and Wang Z 2017 Neural computation 29 2352–2449
[4] Ciregan D, Meier U and Schmidhuber J 2012 2012 IEEE conference on computer vision and pattern recognition (IEEE) pp 3642–3649
[5] Amma C, Gehrig D and Schultz T 2010 Proceedings of the 1st Augmented Human international Conference pp 1–8
[6] Ghahramani Z 2001 Hidden Markov models: applications in computer vision (World Scientific) pp 9–41
[7] Arsalan M and Santra A 2019 IEEE Sensors Journal 19 8855–8864
[8] Xingjian S, Chen Z, Wang H, Yeung D Y, Wong W K and Woo W C 2015 Advances in neural information processing systems pp 802–810
[9] 2020 Motion Sensors Explainer https://www.w3.org/TR/motion-sensors/ [Online; accessed 06-August-2020]
[10] Sherstinsky A 2020 Physica D: Nonlinear Phenomena 404 132306
[11] Hochreiter S and Schmidhuber J 1997 Neural computation 9 1735–1780
[12] Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H and Bengio Y 2014 arXiv preprint arXiv:1406.1078
[13] Graves A and Schmidhuber J 2005 Neural networks 18 602–610
[14] 2020 File:The LSTM cell.png - Wikimedia Commons [Online; accessed 06-August-2020] URL https://commons.wikimedia.org/wiki/File:The_LSTM_cell.png
[15] 2020 File:Gated Recurrent Unit, type 1.svg - Wikimedia Commons [Online; accessed 06-August-2020] URL https://en.wikipedia.org/wiki/Gated_recurrent_unit#/media/File:Gated_Recurrent_Unit_base_type.svg
[16] Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R 2014 The journal of machine learning research 15 1929–1958
[17] Semeniuta S, Severyn A and Barth E 2016 arXiv preprint arXiv:1603.05118
[18] Kingma D P and Ba J 2014 arXiv preprint arXiv:1412.6980
[19] Glorot X and Bengio Y 2010 Proceedings of the thirteenth international conference on artificial intelligence and statistics pp 249–256