Employing an LSTM model with a self-attention mechanism to model variations in running pace

Haoran Jin[1,*], Juntao Wu[2]

(1: Northeast Normal University, Jilin, China

2: Changchun Foreign Languages School, Jilin, China)

ABSTRACT

During the process of running, the pace varies with changes in time. Experienced runners adjust their pace by modifying stride length and frequency. To simulate the relationship between stride length, frequency, and pacing throughout a run using neural network methods, we can leverage the temporal patterns present in the data. The unique memory capabilities of Long Short-Term Memory (LSTM) networks make them well-suited for this task. To enhance the predictive accuracy of the LSTM model, we incorporate a self-attention mechanism that improves the model's ability to associate information from different positions within the input data. This self-attention mechanism enables the model to identify which parts of the data are more significant and thus allocate greater focus during predictions, ultimately enhancing performance. The LSTM method augmented with a self-attention mechanism can accurately fit changes in running pace based on historical exercise data, providing valuable insights for practitioners in running sports. Experimental results indicate that this model achieves a fitting error on the order of 10^{-4}.

Keywords: Neural Network; LSTM; Running pace change fitting; Self-attention mechanism; TOPSIS; Enhancing performance

1 INTRODUCTION

Running, as a sport with relatively low entry barriers and widespread popularity, presents athletes with the important challenge of maximizing training effectiveness. In running, participants adjust their pace based on their current physical condition to achieve their target distance. An optimal pacing strategy not only conserves the runner's energy but also maximizes the benefits of physical training. Cadence and stride length are two critical factors that influence pacing. Investigating the effects of cadence and stride length on pacing holds significant research value. If we can model changes in pace based on a runner's historical data regarding cadence or stride length, it would enable real-time adjustments to these parameters during future workouts. Therefore, there is a pressing need to design a predictive model for running pace that effectively processes time series data.

2 RELEATED WORK

As artificial intelligence rapidly advances, an increasing number of machine learning methods are being employed for prediction and fitting tasks [1]. These include algorithms such

as logistic regression, support vector machines, and neural networks [2][3]. In the realm of sports, several athletic disciplines have adopted machine learning techniques to facilitate relevant predictive tasks. For instance, Urtats Etxegarai et al. utilized machine learning technology to estimate the lactate threshold of recreational runners [4]. Their system modeled lactate evolution using recurrent neural networks, achieving an impressive accuracy rate of 89.52% in estimating the lactate thresholds of athletes while demonstrating high generalization capabilities. Kichang Lee et al. analyzed the impact of automated batting systems (ABS) in professional baseball using data from the Korean KBO League [5]. The ABS employs machine learning, computer vision, and precise tracking technologies to automate batting processes [6]. This study examined pitching data from 2,515 games to compare decisions made by human umpires with those rendered by the ABS. Gyanna Gao et al. leveraged artificial intelligence for fair and accurate classification of tennis skill levels and training stages in a pilot study that processed data from 12 participants using a support vector machine (SVM) algorithm [7] . The model demonstrated an overall accuracy rate of 77% in classifying players as beginners or intermediate-level competitors while maintaining low false positive and false negative rates effectively distinguishing between skill levels. These methodologies illustrate that machine learning has shown promising results across various categories within sports prediction environments; however, there remains a notable gap in research specifically addressing variations in running pace dynamics. Furthermore, existing models tend to overlook how temporal characteristics inherent in time series data can influence predictive accuracy.

Currently, various deep learning models based on increasing the depth of neural networks are being more widely applied to artificial intelligence tasks [8]. Simon Lacan proposed a stacked deep learning model for detecting high-potential soccer players, which demonstrated significantly better results than traditional statistical methods when tested on an open-source database [9] . Among the various deep learning models, Recurrent Neural Networks (RNNs) excel at handling sequential time series data. Long Short-Term Memory (LSTM) networks are a specialized type of RNN that effectively retain long-term memory information and exhibit superior predictive performance compared to standard RNNs. Hsuan-Cheng Sun and colleagues utilized LSTM networks to predict player performance in Major League Baseball, revealing that LSTMs outperform other memory types in terms of effectiveness [10]. Similarly, Rahul Chakwate and his team employed LSTM networks to forecast the final outcomes of cricket matches at any given point during the game [11].

The self-attention mechanism is a significant technique in deep learning that enables models to directly consider the relationships between various positions within a sequence when processing sequential data, without relying on external information [12]. The core idea of this mechanism is to compute the relevance (or weights) of each element in the sequence with respect to other elements and update each element's representation based on these weights. While LSTM networks can learn long-term dependencies, they may struggle to capture distant correlations when dealing with longer sequences. In contrast, the self-attention mechanism allows models to focus on parts of the input sequence that are most relevant to the current output during generation, thereby enhancing both model expressiveness and generalization capability. Combining these two approaches leverages LSTM's long-term memory capabilities alongside the dynamic focusing ability of self-attention mechanisms. Pouya Shaeri et al. proposed a semi-supervised fake news detection method based on

emotional encoding and LSTM combined with self-attention; experiments demonstrated that this approach exhibited superior performance in terms of accuracy, recall rate, and measurement metrics [13]. Similarly, Bingyu Li et al. introduced an interpretable deep learning architecture utilizing both LSTM and self-attention mechanisms for high-accuracy sentiment analysis of text [14]. The integration of self-attention mechanisms with LSTMs effectively harnesses their respective strengths, significantly improving model performance in handling complex sequential data.

During the process of running, factors such as cadence, stride length, and pace continuously change over time or with distance covered. When studying the relationships among these variables, it is essential to fully consider their temporal characteristics while also leveraging prior information effectively. The unique memory capabilities of Long Short-Term Memory (LSTM) networks can adeptly address this task. To enhance the predictive accuracy of the LSTM model, an attention mechanism has been incorporated into the framework to strengthen its ability to associate different positional information within the input data. This self-attention mechanism enables the model to identify which parts of the data are more significant and thus allocate greater focus on these areas during prediction, ultimately improving overall performance. Based on this approach, this paper employs a Self-Attention-LSTM model to fit variations in running pace [15].

3 MATERIALS AND METHODS

3.1 Fundamental Principles of the Self-Attention Mechanism

In neural networks, the attention mechanism enhances model performance by applying weighted processing to input data, guiding the model to focus more on information that is crucial for the current output [16]. This mechanism has been widely extended to various time series processing tasks. The self-attention mechanism is a variant of the attention mechanism; its fundamental principle involves calculating the relevance (or "attention weights") between each element in a sequence and all other elements. These weights are then utilized to update the representation of each element, ensuring that every representation incorporates information from other elements within the sequence. This approach effectively addresses long-range dependency issues and offers higher computational efficiency, as illustrated in Figure 1:

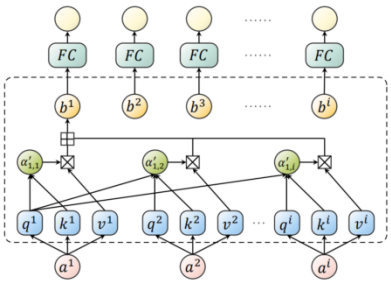

Fig. 1: Diagram of the Self-Attention Mechanism Structure.

The variable a^i in the figure represents the feature values of the input vector, while b^i denotes the output vector. The calculation formula for b^1 is as follows:

$$b^1 = \sum_i \alpha'_{1,i} v^i, \tag{1}$$

The terms q, k, and v refer to the Query values, Key values, and Value values respectively. Unlike traditional attention mechanisms, in this context, q, k, and v are derived from the same input. The calculation formula is as follows:

$$q^i = w^{q_i} \cdot a^i, \qquad k^i = w^{k_i} \cdot a^i, \qquad v^i = w^{v_i} \cdot a^i, \tag{2}$$

In this context, $w^{q_i}, w^{k_i}, w^{v_i}$ are the weight parameters that need to be learned.

3.2 Long Short-Term Memory Neural Network (LSTM)

The Long Short-Term Memory neural network was first introduced by Hochreiter and Schmidhuber in 1997 [17]. It is fundamentally a specialized type of recurrent neural network [18]. LSTM incorporates three distinct "gates"—the forget gate, input gate, and output gate—to regulate the retention and updating of information. This mechanism enables LSTMs to effectively capture long-term dependencies within sequential data, making them particularly suitable for processing and predicting significant events characterized by extended intervals and delays in time series data. The structure of an LSTM unit is illustrated in Figure 2:

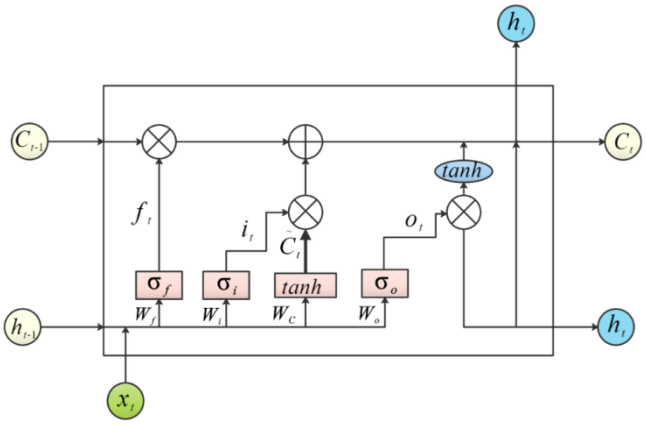

Fig. 2: Diagram of the LSTM Structure.

The forget gate is responsible for determining which information to discard. This gate reads the value of the previous hidden state h_{t-1} and the current input x_t, then passes this information through a sigmoid function. It outputs a value between 0 and 1 for each element in the previous cell state C_{t-1}. A value of 1 indicates "completely retain," while a value of 0 signifies "completely discard."

$$f_t = \sigma_f(W_f \cdot [h_{t-1}, x_t] + b_f), \tag{3}$$

In this context, W_f represents the weights of the forget gate, b_f denotes the bias of the forget gate, σ_f refers to the sigmoid activation function, and f_t indicates the output of the forget gate.

The input gate is responsible for updating the cell state. This process consists of two parts: first, a "input gate layer" using a sigmoid function determines which values will be updated; second, a tanh layer generates a new candidate value vector $\tilde{C}_t$ to be incorporated into the state.

$$i_t = \sigma_i(W_i \cdot [h_{t-1}, x_t] + b_i), \quad (4)$$

$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C), \quad (5)$$

In this context, W_i and W_C represent the weights of the input gate, while b_i and b_C denote the biases associated with it. The symbol σ_i refers to the sigmoid activation function, it indicates the output of the input gate, and $\tilde{C}_t$ signifies the candidate cell state.

The second step involves multiplying the previous state by f_t, discarding the information that needs to be omitted. Subsequently, we add $i_t * \tilde{C}_t$ to obtain the new cell state:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t, \quad (6)$$

The term C_t refers to the most recent cell state at time t, while the symbol $*$denotes the element-wise multiplication operation.

The output gate is responsible for producing the information output. It determines the portion of the cell state to be output through a sigmoid layer, and ultimately derives the final output value via a dot product operation.

$$o_t = \sigma_o(W_o \cdot [h_{t-1}, x_t] + b_o), \quad (7)$$

$$h_t = o_t * tanh(C_t), \quad (8)$$

The variable h_t represents the output value at time step t, while W_o and b_o denote the weights and biases of the output gate, respectively. Finally, both the output h_t and the cell state C_t are passed to the next unit, where the aforementioned computational process is repeated.

3.3 Self-Attention LSTM Structure

The structure of the Self-Attention-LSTM neural network is illustrated in Figure 3, consisting of a total of four layers within its architecture:

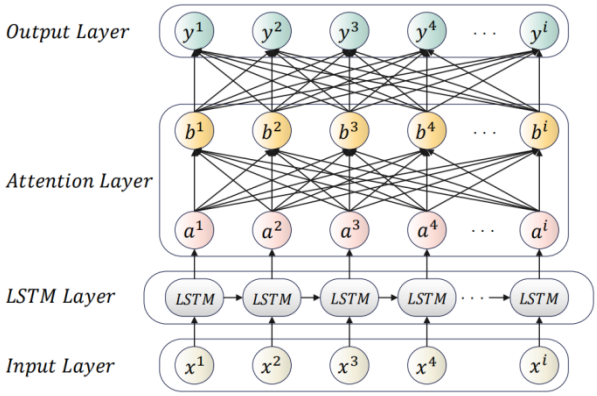

Figure 3: Diagram of the Self-Attention LSTM Structure.

(1) Input Layer: This layer preprocesses the data related to step frequency, stride length, and pace. It addresses any missing values and calculates the average values to generate input data for the neural network. If the length of the data is denoted as i, then the input vector can be represented as $X = [x^1, x^2, x^3, \cdots, x^i]^T$.

(2) In the LSTM layer, the input vector $X = [x^1, x^2, x^3, \cdots, x^i]^T$ is processed according to the operational principles of the LSTM layer. This process enables the model to learn the relationships among various indicators and produces an output vector $a = [a^1, a^2, a^3, \cdots, a^i]^T$.

(3) In the Attention layer, data is processed in a manner illustrated in Figure 1, where the internal relationships among the elements of the learning metrics are computed repeatedly. Ultimately, this results in an output represented as $b = [b^1, b^2, b^3, \cdots, b^i]^T$.

(4) The output layer, represented by the vector $b = [b^1, b^2, b^3, \cdots, b^i]^T$, performs the final computation through a fully connected output layer. This process facilitates the learning tasks of input pacing and fitting for both cadence and stride length. Consequently, the output is obtained as $Y = [y^1, y^2, y^3, \cdots, y^i]^T$.

3.4 Model Loss Function

In this paper, we utilize MSE loss function to assess the model's fitting performance. The calculation formula is as follows:

$$\text{Loss function} := \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y^i)^2, \tag{9}$$

In this context, $\hat{y}_i$ represents the true data, while y^i denotes the fitted data. A smaller Loss function indicates that the model's fit to the actual values exhibits a lower degree of error, thereby reflecting a more accurate fitting performance.

4 RESULTS AND DISUSSION

The subject of this study is a runner's records from three 5km runs. Utilizing this dataset, we aim to extract relevant running information about the athlete and further enhance the generalizability of our model. Through data preprocessing, we obtained metrics such as average cadence, average stride length, and average pace, resulting in a total of 100 data points. We selected 50 data points for training purposes and reserved another 50 for testing.

4.1 Data Preprocessing

Different features have varying units and significant differences in magnitude. Therefore, we first preprocess the data by calculating the average values for each group of features. We then determine the proportion of individual data points relative to their respective group averages. This process allows us to normalize the data within a consistent range, resulting in a scatter plot that illustrates the distribution of feature quantities over time (in kilometers).

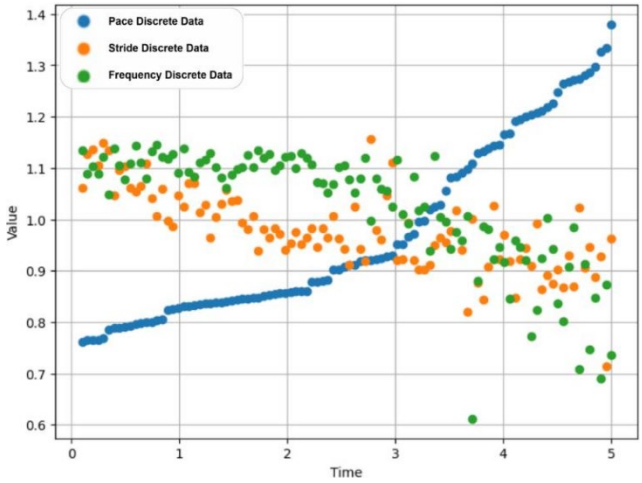

Fig. 4: Scatter Distribution Diagram of Various Features.

Figure 4 shows the variation of different variables over time, with blue dots representing pace points, orange dots representing stride length points, and green dots representing cadence points. Through a certain processing, all three are displayed on one graph, but their units are obviously different. The values on the horizontal axis represent different kilometers, with the unit being km. Different kilometers correspond to different times.

The interpolation method is employed to transform the scatter plot into a continuous function graph. Additionally, the Savitzky-Golay filter is utilized to obtain a smoother representation of the function image. In this process, the filter's window length is set to 30, and a third-order polynomial fitting is applied. The signal extension type for the filter is selected as "nearest."

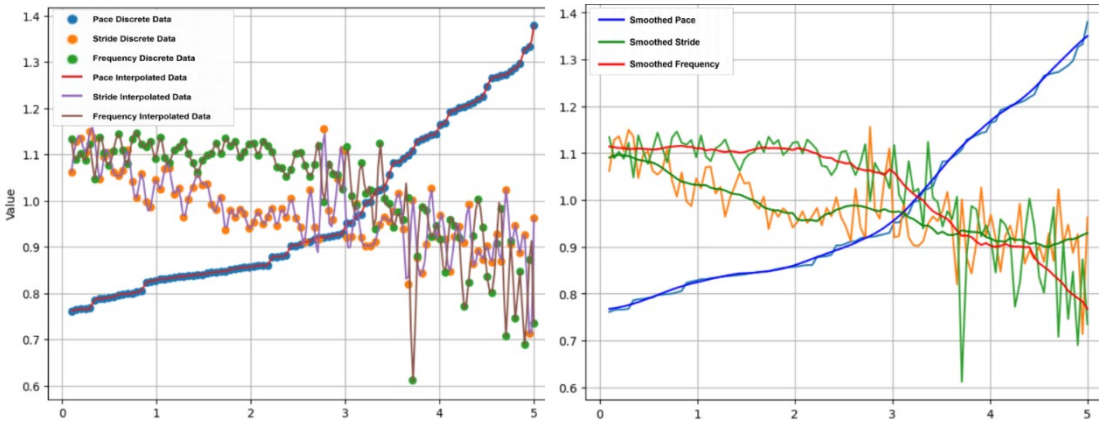

Fig. 5: (a). Perform interpolation on the scatter plot. (b). Refining the interpolation function with a filter.

4.2 Tuning of Hyperparameters

The following section presents the hyperparameters related to the experimental process:

1) Optimizer: The model employs the Adam optimizer. This optimizer has demonstrated excellent performance across various scenarios, with a learning rate set at 0.01.

2) LSTM Layers: Typically, the number of LSTM layers is chosen to be between 1 and 2.

Through experimentation, it was determined that setting the LSTM layer count to 1 achieves satisfactory fitting results.

3) Batch Size and Time Steps: In neural network training, batch processing is more efficient than single-instance training; thus, a batch size of batch_size = 5 is utilized. The temporal data points are segmented based on distance traveled, with a step size of 0.05 and a total length of 5 units, resulting in a total of 100 data points.

4) Comparative Model Structural Parameters: This study compares the fitting performance of Self-Attention-LSTM, conventional LSTM, and fully connected neural networks during both training and fine-tuning processes. The structural parameters for these comparative models are as follows:

Table 1: Comparison of Model Structural Parameters.

Conventional LSTM	MLP Neural Network
layer = 1	layer = 1
hidden_size = 1	hidden_size = 16
output_size = 1	output_size = 1

5) Iteration Counts: This paper includes two types of iteration counts, namely max_epoch and min_epoch, which correspond to the training on the training set and fine-tuning during testing, respectively. Based on experimental results, it is recommended that the values for these iteration counts be set as follows: max_epoch = 5000 and min_epoch = 100.

4.3 Refinement of the Self-Attention LSTM Model Fitting Performance

The model utilizes 50 data points as input, where the Self-Attention layer is responsible for learning the sequential features of stride and its own characteristics. Meanwhile, the LSTM layer focuses on understanding the relationship between stride frequency and pace. Upon completion of training, a fitted graph is produced. Figure 6 illustrates the fitting performance of the Self-Attention-LSTM model:

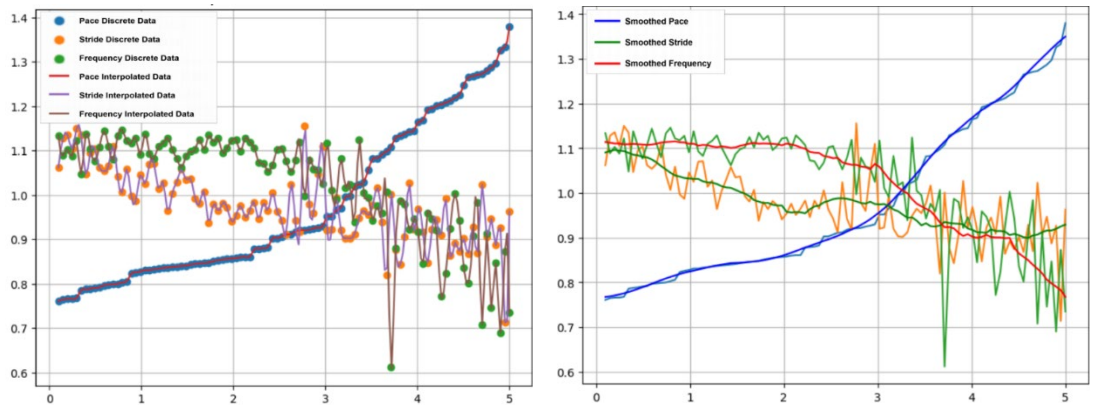

Fig. 6: (a). Fitting the Relationship Between Stride and Pace in the Self-Attention LSTM Model. (b). Fitting the Relationship Between Frequency and Pace in the Self-Attention LSTM Model.

The model requires only the input of stride length or cadence to automatically calculate the corresponding pace results. Due to the properties of the function, if the running pace is

known, it can also quickly deduce the relevant cadence and stride length.

4.4 The fitting performance of the conventional LSTM model and the MLP model

Under identical experimental conditions, a comparative study was conducted using the LSTM model and the MLP model. The results are illustrated in Figure 7.

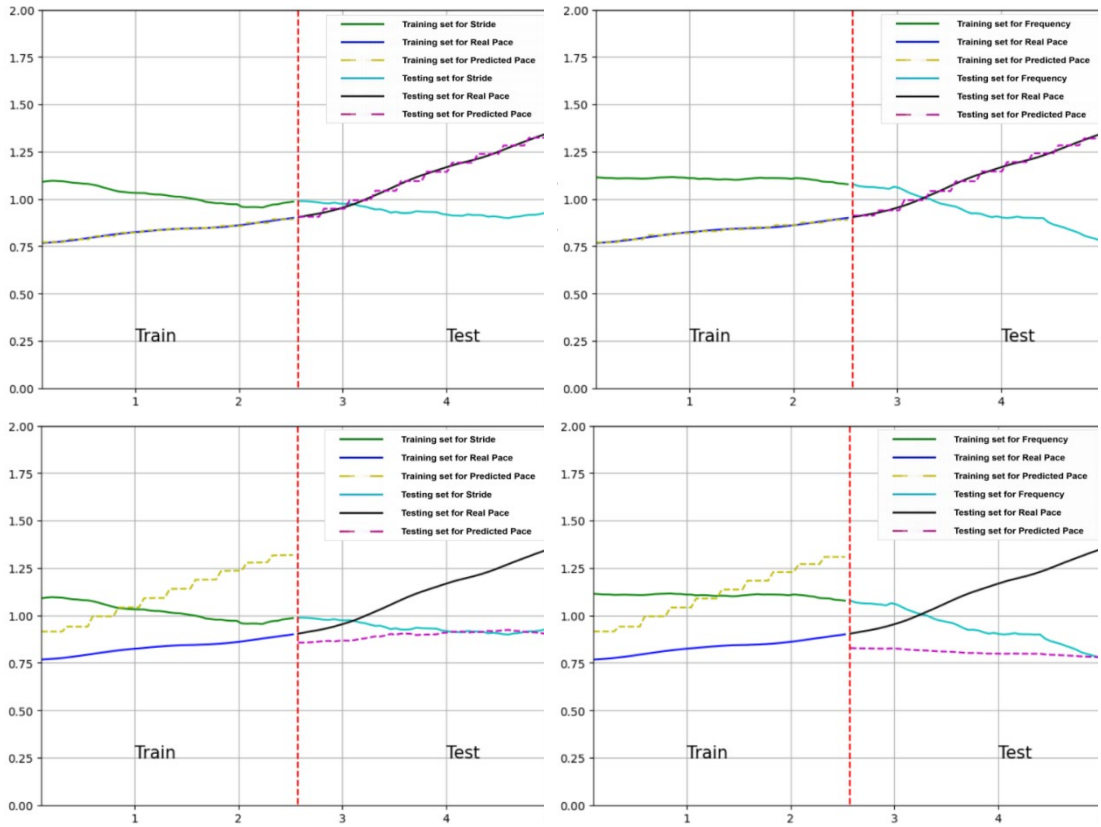

Fig. 7: (a)(b). The results of conventional LSTM Model. (c)(d). The results of MLP Model.

The two upper images in Figure 6 illustrate the fitting performance of the LSTM model, while the two lower images depict the fitting results of the MLP model. After conducting an equal number of fine-tuning iterations, it is evident that the conventional LSTM model achieves a satisfactory fitting performance on this dataset. In contrast, the MLP neural network demonstrates subpar fitting results, with significant deviations between its fitted output and the actual data.

4.5 Comparison of Errors Among Three Models

Table 2 and Table 3 present the final training and testing errors for pace fitting based on three different models of stride length and frequency, respectively.

Table 2: Errors in Stride and Pace Across Three Models.

Train accuracy	Test accuracy
Self-Attention-LSTM: 2.07×10^{-5}	Self-Attention-LSTM: 2.15×10^{-4}
LSTM: 3.33×10^{-5}	LSTM: 2.72×10^{-4}

MLP: 1.73×10^{-4}	MLP: ---

Table 3: Errors in Frequency and Pace Across Three Models.

Train accuracy	Test accuracy
Self-Attention-LSTM: 3.10×10^{-5}	Self-Attention-LSTM: 2.20×10^{-4}
LSTM: 5.54×10^{-5}	LSTM: 4.35×10^{-4}
MLP: 2.01×10^{-3}	MLP: ---

Based on the experimental results, it was found that the Self-Attention-LSTM model exhibits a fitting performance comparable to that of the conventional LSTM model, with a slight advantage for the Self-Attention-LSTM. In contrast to the MLP model, the Self-Attention-LSTM significantly enhances training effectiveness on the training set, and its fitted images on the test set align more closely with real data.

5 CONCLUSION

This study investigates the application of a Self-Attention LSTM model to fit variations in running pace, utilizing metrics such as cadence, stride length, and pace to learn feature relationships. The model takes cadence, stride length, and pace as inputs and employs self-attention mechanisms to capture the intrinsic relationships within the feature sequences. Additionally, it utilizes LSTM methods to understand inter-feature relationships, thereby establishing correspondences among features that enable real-time feedback on data values at any given moment. Experiments conducted using 5 km running data yield the following conclusions:

(1) The Self-Attention-LSTM is well-suited for processing time series data. This model integrates the advantages of both methodologies, allowing for improved learning of relationships between data points compared to traditional fully connected networks.

(2) When comparing Self-Attention-LSTM with conventional LSTM methods, the former demonstrates superior performance. This advantage arises from the self-attention mechanism's ability to consider direct connections between different parts of a sequence during processing rather than relying solely on sequential information transfer. Such a global perspective enhances the model's understanding of complex structures inherent in sequence data.

(3) The accuracy achieved by employing Self-Attention-LSTM for fitting variations in running pace is notably high, with errors reaching an order of magnitude around . This approach can be generalized across various running scenarios and offers valuable insights for training practices in running.

6 ACKNOWLEDGEMENTS

The author is grateful for the support of the "The National Scientificand Technological Innovation Talent Training Program" project, and for the useful discussion by Professor Shuguan Ji and Professor Jian Zu from the School of Mathematics and Statistics of Northeast Normal University.

REFERENCES

[1] Zhi-Hua Zhou. 2021. Machine Learning. Springer Singapore.

[2] Starbuck, C. 2023. Logistic Regression. In: The Fundamentals of People Analytics. Springer, Cham. https://doi.org/10.1007/978-3-031-28674-2_12

[3] Mingchen Li and Zhiji Yang. 2022. Deep Twin Support Vector Networks. In Artificial Intelligence: Second CAAI International Conference, CICAI 2022, Beijing, China, August 27–28, 2022, Revised Selected Papers, Part III. Springer-Verlag, Berlin, Heidelberg, 94–106. https://doi.org/10.1007/978-3-031-20503-3_8.

[4] Etxegarai U, Portillo E & Irazusta J, et al. 2018. Estimation of lactate threshold with machine learning techniques in recreational runners. Applied Soft Computing 63: 181–196.

[5] Lee K, Han K & Ko J. 2024. Analyzing the impact of the automatic ball-strike system in professional baseball: A case study on kbo league data. ArXiv: 2407.15779.

[6] Chai J, Zeng H & Li A, et al. 2021. Deep learning in computer vision: A critical review of emerging techniques and application scenarios. Machine Learning with Applications 6: 100134.

[7] Gao G, Liao H Y & Hu Z. 2024. Ai for equitable tennis training: Leveraging ai for equitable and accurate classification of tennis skill levels and training phases. ArXiv: 2406.16987.

[8] Krizhevsky A, Sutskever I & Hinton G E. 2012. Imagenet classification with deep convolutional neural networks. Communications of the ACM 60: 84 - 90.

[9] Lacan S. 2024. Stacking-based deep neural network for player scouting in football. ArXiv: 2403.08835.

[10] Sun H C, Lin T Y & Tsai Y L. 2022. Performance prediction in major league baseball by long short-term memory networks . ArXiv: 2206.09654.

[11] Chakwate R U. 2020. Analyzing long short term memory models for cricket match outcome prediction. International Journal for Research in Applied Science and Engineering Technology 8(11): 1–8.

[12] Vaswani A, Shazeer N & Parmar N, et al. 2023. Attention is all you need. 2023. ArXiv: 1706.03762.

[13] Shaeri P & Katanforoush A. 2024. A semi-supervised fake news detection using sentiment encoding and lstm with self-attention. ArXiv: 2407.19332.

[14] Li B, Zhang D & Zhao Z, et al. 2024. Quantum-inspired interpretable deep learning architecture for text sentiment analysis. ArXiv: 2408.07891.

[15] Lai G, Chang W C & Yang Y, et al. 2018. Modeling long- and short-term temporal patterns with deep neural networks. ArXiv: 1703.07015.

[16] Bahdanau D, Cho K & Bengio Y. 2016. Neural machine translation by jointly learning to align and translate. ArXiv: 1409.0473.

[17] Hochreiter S & Schmidhuber J. 1997. Long short-term memory. Neural computation 9: 1735-80.

[18] Lipton Z C, Berkowitz J & Elkan C. 2015. A critical review of recurrent neural networks for sequence learning. ArXiv: 1506.00019.