High-Performance Stacked Ensemble Model for Stride Length Estimation with Potential Application in Indoor Positioning Systems

Hang T. Duong[1,2], Kha M. Hoang[1], Hung V. Pham[3], and Vu A. Trinh[2]
[1] Faculty of Electronics, Hanoi University of Industry, Hanoi, Vietnam
[2] University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam
[3] Trusting Social, Havana Centre, Ho Chi Minh City, Vietnam
Email: hangdt@haui.edu.vn; khahoang@haui.edu.vn, hungpv.esoft@gmail.com, anhvutrinh1811@gmail.com

Abstract —The pedestrian dead reckoning (PDR) infrastructure-independent positioning method is of particular interest for many researchers due to its effectiveness in indoor positioning systems. The PDR technique consists of three main components including stride detection, movement heading estimation and stride length estimation (SLE). Among those, SLE results can be utilized in numerous applications, such as indoor positioning, disease diagnosis, incident detecting for patient warning, etc.. Traditional solutions for estimating stride length usually have simple structures with short calculation time but they do not ensure the expected accuracy as desired. To enhance the estimated stride length, recently, many solutions employing deep learning based techniques such as Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM), have been developed to improve the SLE result. Although the accuracy in SLE has been improved compared to traditional methods, these deep learning based solutions still remain some limitations, such as complex structure and huge training parameters. This paper proposes some solutions of using machine learning based stacked ensemble model and simple machine learning algorithm to improve the accuracy of SLE while satisfying requirements such as simple structures, short execution time. The experimental results on real field data show that the proposed approaches outperform other state-of-the-art deep learning based methods on both SLE accuracy and computation time by at least 12% more accurate and seven times faster, respectively.

Index Terms—Indoor Positioning System, Linear regression, Machine learning, Pedestrian Dead Reckoning, Stacked ensemble model, Stride length estimation

I. Introduction

In the past decades, indoor positioning has been used in many applications that serve our life such as assisting elderly management in nursing homes, finding a safe exit in an emergency, warehouse management, or navigation in huge shopping malls. While GPS has represented remarkable achievements in applications in outdoor environments, the GPS signal is often blocked inside buildings. Therefore, indoor positioning still remains a huge challenge for researchers to achieve the desired performance due to the frequent change of environment, movement of people, etc.. Because of the wide applicability of indoor positioning, many researchers have shown their particular interest in finding the best solutions for indoor localization.

One of the first indoor positioning systems, RADAR system, was introduced by Microsoft Research Asia [1] in 2000. In 2005, the University of Maryland built the Horus system [2]. The systems developed in recent years (2015) can be mentioned as LIFS system of Tsinghua University [3]; The WHERE @ UM system of Minho University [4] has made remarkable improvements compared with previous research.

In [5]-[7], authors have shown that the individual positioning methods achieve less accurate than integrated methods. Recently, with the increasingly robust development of microelectromechanical systems (MEMS), most smartphones have been equipped with inertial sensors such as accelerometers, gyroscopes and magnetic sensors which then were utilized to develop indoor positioning system. The PDR is the most popular technique which stimulated much interest because it does not either depend on infrastructure or need a prior training process. The main components of the PDR are stride detection, stride length estimation, and movement heading estimation. Previous publications [1-6] have presented many effective solutions for detecting the step and estimating the movement direction. For stride length estimation, although several techniques have been presented, it is still lack of efficient methods which satisfy both high accuracy and low computation cost. Therefore, in order to improve PDR indoor positioning, many researchers are now focusing on improving the accuracy of stride length estimation while keeping the computation cost at a reasonable level.

The stride length estimation methods presented in [8]-[11] have simple models which require short computation time but the accuracy is not as good as machine learning based methods as presented later in this section. These methods use signals from accelerometer and gyroscope sensors. However, smartphone Inertial Measurement Unit (IMU) sensors often contain large errors that only allow estimating step lengths with low-level accuracy performance. To improve the efficiency of stride

Manuscript received January 14, 2022; revised July 14, 2022.
doi:10.12720/jcm.17.8.652-660

estimation, it is usually necessary to create extra features to get specific characteristics that easily distinguish the signals. Those approaches have been researched in state-of-the-art studies [12]-[15].

Recently, many researchers have employed deep learning to estimate stride length with the aim to improve system efficiency. In [14], a convolutional neural network was proposed to estimate a pedestrian's stride length to detect and classify footstep abnormalities in order to diagnose the disease in some patients. In [15], the authors used complex deep learning networks combined with auto-encoder techniques to filter out the noise of raw signals obtained from sensors on the phone. New features are generated from the raw data to improve the accuracy in stride length estimation. In [16], the authors combined multiple models to recognize five modes (calling, handheld, pocket, armband, and swing), and proved these methods independent on how to bring the phone. In other words, the location of the phone does not affect the accuracy of steps length estimation. In this paper, authors constructed high features based on Weinberg [8], Kim [9], Ladetto [10] and Scarlett [11] to estimate stride length using deep learning model. However, the common problem of the methods which use deep learning network is that the structure of the network models is complex; the training time is long due to huge parameters needed to be trained.

This paper aims at proposing stride length estimation solutions to achieve both the simplicity of the model as well as the desired accuracy. First, we have utilized Linear Regression model which ensures to maintain accuracy and simple model structure with less parameters to be trained. Second, we have proposed a stacked ensemble model. This method significantly improves the estimation accuracy while ensuring that the model is not too complicated and the estimated number of parameters is small enough. The experimental results show that the proposed methods have achieved more accurate stride length estimation than other methods which used deep learning networks on the same data set. For the purpose of ensuring generality and objectivity, the proposed solutions are verified on the benchmark public dataset. (https://github.com/Archeries/StrideLengthEstimation/tree/master/Benchmark-Dataset-for-Adaptive-Stride-Length-Estimation) provided by [16].

The next sections of this article are organized as follows: Section II presents some studies related to SLE. Section III describes our proposed method. The results and discussions are presented in section IV. Section V is the conclusion of the paper.

II. Related Work

This section presents traditional methods as well as modern methods to explore studies related to stride length estimation. Traditional methods normally have simple structures and are suitable for real-time applications that do not require high accuracy and can be performed on low-resource devices such as smartphones. On the other hand, the deep learning based methods often require very powerful processor to be able to apply for real-time applications.

A. Traditional Pedestrian Dead Reckoning

Fig. 1 illustrates the block diagram of a traditional pedestrian dead reckoning system utilizing signals obtained from inertial sensors on smart phones. These signals were processed, filtered noise, and then fed into step length estimator.

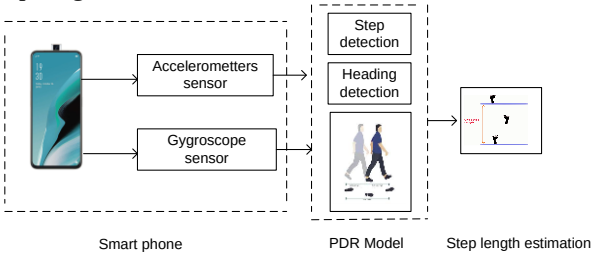

Fig. 1. A traditional step length estimation block diagram.

Techniques presented in [8]-[11] used only raw data reported from inertial sensors on a smartphone to estimate stride. Some previous publications, such as Weinberg [8], Kim [9], Scarlett [11], estimated stride length according to (1), (2), and (3):

$$L_{\text{Weinberg}} = K\sqrt[4]{a_{\max} - a_{\min}} \tag{1}$$

$$L_{Kim} = K\sqrt[3]{\frac{\sum_{i=1}^{N}|a_i|}{N}} \tag{2}$$

$$L_{Scarlett} = K\frac{\frac{\sum_{i=1}^{N}|a_i|}{N} - a_{\min}}{a_{\max} - a_{\min}} \tag{3}$$

In (1), (2), (3), K is a experimentally determined parameter, representing the relationship between the real values and the predicted values. $a_{\max}$ and $a_{\min}$ are the maximum and minimum values taken according to the z-axis of the accelerometer at each stride length [8]. In (2) and (3), $|a_i|$ represents the measured acceleration samples in each step and N represents the number of samples corresponding to each step. Meanwhile, Ladetto [10] utilizes the linear relationship among stride frequency, variance of acceleration and weights of the features according to equation (4):

$$L_{\text{Ladetto}} = \alpha \times f + \beta \times v + \gamma \tag{4}$$

where f *is* stride frequency and v denote acceleration variance of α , α and β denote the weights of the features, γ represents a constant which includes the noise.

The above-mentioned methods are relied mainly on the maximum, minimum, or mean acceleration amplitude

without taking into account position of the smartphone. Weinberg [8] is considered to have the best accuracy in estimating the step length compared to the other basic solutions. In this method, the maximum and minimum of acceleration measuring values on the z-axis are used to estimate stride length. Kim [9] proposed an experimental model for determining the step length using the average value of the amplitude of acceleration in each stride, analyzing the vertical and lateral acceleration of the foot while walking. A new integrated method of gyroscope and magnetic compass was used to determine stride by analyzing the relationship between stride and step cycle. In [11], an algorithm was held up appropriately for a variety of subjects and paces, with all variations. However, this algorithm is so simple that it can be tricked when the pedestrian is stationary, i.e., not moving. For instance, a displacement is still estimated although the pedestrian steps at the same position. In [10], Ladetto calculates the stride length based on linear regression machine learning algorithm and the linear relationship among step length, frequency, and the local variance of acceleration amplitude for the purpose of analyzing everyday human activities. The above methods are based only on the maximum and minimum of the acceleration amplitude, the horizontal and vertical mean values of the acceleration amplitude, without considering the direction and position of the device.

These methods are typical studies of the stride estimation problem used in PDR with simple algorithms. The common disadvantage of these methods is the undesirable accuracy, which can be erroneous in some special cases such as jogging, climbing stairs, changing speed while walking or stationary subject.

Recently, there have been a lot of researches aiming at improving the accuracy of stride length using deep learning methods. We will present a deep learning network to estimate stride length in Section B.

B. Deep Learning Based Step Length Estimation

Many studies are suggested to improve the accuracy of stride length estimation based on signals obtained from inertial sensors [12], [13], [17]-[22]. The solutions given in section II.A have partly satisfied all the requirements of estimating stride length in simple applications. However, to improve the accuracy of stride length estimation and solve vital problems in traditional algorithms, solutions using deep learning networks in stride length estimation have been proposed in [14-16], [23]-[27]. Two of them were rebuilt in this article to evaluate the effectiveness of the proposed method compared with the current advanced studies. Fig. 2 depicts the structure of a deep learning network using CNN to estimate the stride length presented in [14]. The deep learning model would normally have inputs of accelerometer and gyroscopes data as the main features. In this diagram, the initial signals are taken from inertial sensors. Raw signals are usually pre-processed and standardized before feeding to the deep learning networks. To rebuild model in [14], we have used CNN algorithm with parameters like the input data of length L0 = 256 with N0 = 6 channels. On the first layer of the network, we choose to learn N1 = 32 filters of length and L1 = 30 samples. After that, on the second layers, the number of filters is N2 = 64 and L2 = 15. The fully connected layer has Nfc = 1024 nodes (details in [14]).

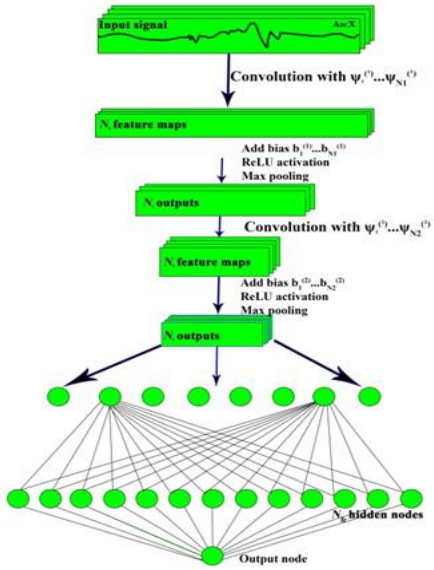

Fig. 2. Diagram of deep learning network structure and estimating general SLE.

The results of stride length estimation using CNN in [14] opens a new approach which used deep learning networks in estimating stride. [14-16], [23-27] is a series of researches using deep learning network to estimate stride length. These studies have improved the accuracy of stride length estimation compared to the original researches by using a combination of individual modern methods. Fig. 3 shows the architecture and the parameters of the model using LSTM [16].

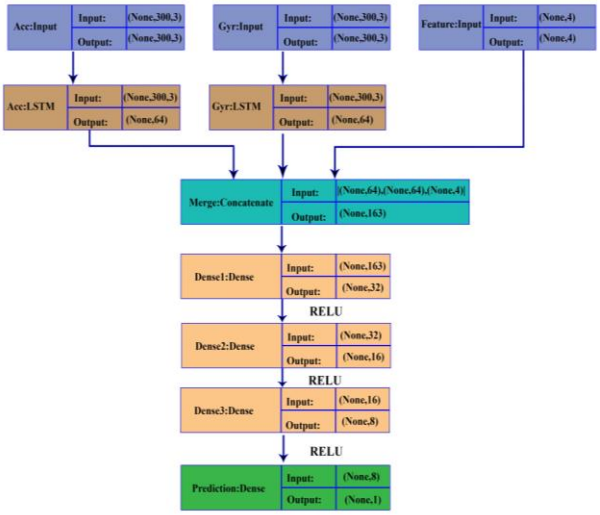

Fig. 3. The architecture and parameter of the LSTM.

In [16], the authors have proposed a stacked ensemble model term Tap-line including a long short term memory module and de-noising auto-encoders to automatically

estimate pedestrian stride to then automatically cancel the sound and eliminate noise in raw inertial sensor data. This approach can improve the accuracy of the SLE estimation with the cost of high computation. In this proposal, authors constructed stride data using the six channels of raw inertial information and the four high–level features, these extra features were obtained by using the models presented in [8]-[11]. By combining those deep learning networks, the results show that the relative errors in [16] are smaller than those achieved by the Weinberg, Kim, Scarlett and Ladetto algorithm.

However, there are some shortcomings of those approaches based on machine learning as follows:

- The architecture of deep learning networks is so complex [14]-[16], [24]-[27].
- The number of parameters to be trained is huge (There are 3,249,889 parameters [14] and 39,713 parameters [16] in CNN and LSTM, respectively)
- The training time is long [14]-[16], [24]-[27].
- It is not suitable for implementing in low resource systems such as smartphones or computer without GPU.

In order to address these challenges, this paper proposes a solution to improve the accuracy of stride length estimation while reducing the complexity of deep learning models. Our solution will be discussed in details in the next section.

III. Proposed Approach

As mentioned in section II.A, traditional methods can be used for applications that do not require high accuracy of SLE and powerful execution systems. For applications requiring high precision, the system can be deployed on a high-performance computer system using the models in section II.B. Proposals in Section II so far are able to solve only one of the two main targets in SLE that are high accuracy and low computation cost. In this paper, a proposal was introduced to satisfy simultaneously unsolved shortcomings in part 2 of high precision, simple structure, not requiring a high-configuration computer system. Therefore, this method meets execution requirements on devices with limited hardware resources such as smartphones. Our proposed model was illustrated in Fig. 4. Signal was received from x-axes, y-axes, and z-axes of inertial sensors on smartphones (accelerometer and gyroscope).

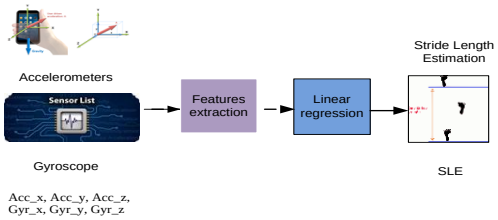

Fig. 4. Architecture of linear regression model for SLE.

The paper presents two main proposals as follows:

- High level feature creation based on the raw data and a linear regression model for SLE.
- Machine learning-based stacked ensemble model including linear regression and k-NN using the high level features for SLE.

A. Linear Regression based Stride Length Estimation

1) System block diagram

We used the data set on the provided link in [16]. The feature extractions are generated before being used to train the models as well as determine the stride length in the estimation phase. The extraction features are computed as shown in Table I, details are presented in subsection III.A.3.

In this proposal, the six raw features are observed from the accelerometer and gyroscope sensors on smartphones. Using these features, we create 10 extraction features per each raw feature, resulting in total of H=60 features. By using the extra features as training input for the linear regression model, mean absolute error is calculated by (10) as presented in section IV.

2) Linear regression

Linear regression is a simple machine learning method as presented in equation (5).

$$y_{LRj} = \omega_1 x_1 + \omega_2 x_2 + ...\omega_H x_H + \omega_0 \quad (5)$$

In equation (5), y_{LRj} is the stride length that is estimated at j-th data point of the i-th stride. Let $\mathrm{W} = [\omega_0, \omega_1, \omega_2, ...\omega_H]^T$ be the coefficient column vector that should be optimized by the training procedure, and $\bar{x} = [1, x_1, x_2, ..., x_H]$ is the input row vector of the extraction data. Then (5) can be rewritten as (6):

$$y_{LRj} = \bar{x}.\mathrm{W} \quad (6)$$

The values of x_h $(h = 1..H)$ are the extra features taken from inertial sensors (accelerometer and gyroscope sensors), H is the number of extraction high level features. The i-th stride length can be estimated by (7):

$$y_{LRi} = \frac{1}{J}\sum_{j=1}^{J} y_{LRj} \quad (7)$$

In which, y_{LRi} is the stride length that is estimated at the i-th stride. J is the number of samples (data points) in each stride.

3) Extention feature

Using extra features during model training helps to identify and classify data clearly.

Compared with the scenario where only the main features are used, if adding high level features, the system will extract the outstanding features that represent the output to be estimated. For example, in Table I, the extra features F1, F3, F4, F9, and F10 represent the mean, minimum, maximum, the first, and the last values of the measured data of a stride length. These values will be

relatively different between stride lengths. However, using only these features might not guarantee the typical characteristics of the data. Therefore, in addition to these features, we also compute other features such as the standard deviation, the variance of the data at each step resulting in F2 and F7, respectively. Adding this feature helps estimation be easier and noise is eliminated. Based on the above basis, we added F5, F6, F8 extra features to make it easy for the system to identify how one stride length differs from the other.

TABLE I: FEATURES EXTRACTION.

Feature index	Definition
F1	The mean absolute values of acceleration and Gyroscope of the X-axis, Y-axis, Z-axis are defined as $\mu_a = mean(\lvert a_{ia} \rvert)$, $\mu_g = mean(\lvert a_{ig} \rvert)$
F2	Std - Standard deviation of groups, $STD = \sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2 / (n-1)}$ x_i : Value of the i-th point in the data set $\bar{x}$: The mean value of the data set n: The number of data points in the data set
F3	$Min = \min(a_{ia})$, $Min = \min(a_{ig})$ Minimum value of data group
F4	$Max = \max(a_{ia})$, $Max = \max(a_{ig})$ Maximum value of data group
F5	$skew_a = E\left[\left((a_{ia} - mean(a_{ia})) / \sqrt{var(a_{ia})}\right)^3\right]$ $skew_g = E\left[\left((a_{ig} - mean(a_{ig})) / \sqrt{var(a_{ig})}\right)^3\right]$
F6	Kurt: $kur_a = E\left[\left((a_{ia} - mean(a_{ia})) / \sqrt{var(a_{ia})}\right)^4\right]$ $kur_g = E\left[\left((a_{ig} - mean(a_{ig})) / \sqrt{var(a_{ig})}\right)^4\right]$
F7	Var: The acceleration variance, $\sigma_a^2 = var(a_{ia})$ The gyroscope variance, $\sigma_g^2 = var(a_{ig})$
F8	sem: Standard error of the mean of groups, $\sigma_a = \sigma / \sqrt{N}$ $\sigma_g = \sigma / \sqrt{N}$ σ_a , σ_g - standard error of the mean σ - the Standard deviation of the original distribution $\sqrt{N}$ - square root of the sample size N - sample size
F9	first: Compute first of group values
F10	last: Compute last of group values

B. Machine-Learning Based Stacked Ensemble Model

1) System block diagram

In Fig. 5, a machine learning-based stacked ensemble model (a combination of k-NN and Linear Regression) has been proposed. The k-NN algorithm will be presented in details in subsection B.2. k-NN is the most basic machine learning algorithm in the classification problems. In this proposal, we chose the initialization value for k is 10. We used the extra features generated as presented in subsection A.3 as training input for the linear regression model and the k-NN model. After training the two models, the outputs of the two models are fed into a linear function model of the stacked ensemble model. The weights of this model are calculated based on a simple linear regression algorithm. The estimated stride length is the linear function output.

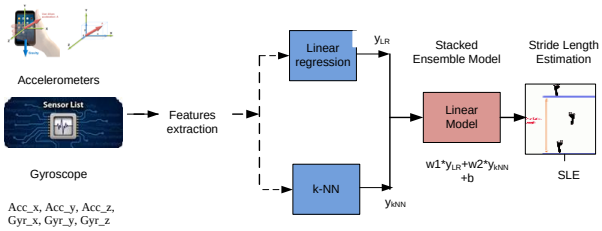

Fig. 5. Architecture of stacked ensemble model

2) K- nearest neighbor

In this paper, k-NN algorithm is used to find k nearest neighbors in the training data compared to the input data for SLE. To obtain results, this algorithm is mainly based on the k closest neighbors using the Euclidean distance criterion according to (8):

$$d_j = \sqrt{\sum_{h=1,}^{H} \left(x_{jh} - x_{th}\right)^2} \qquad (j = 1 \div J) \tag{8}$$

where x_{jh} and x_{th} are the i-th elements of the j-th training data point and the test data point, respectively. H is the number of extra features. J is the number data point (strides) of training data

Using the set P of the index of k training data points which is chosen among the training data with the smallest distances d_j, let l_j be the stride length of the corresponding d_j. The stride length ($y_{k\text{-}NN}$) of test data can then be computed as (9):

$$y_{k-NN} = \frac{1}{k}\sum_{p\in P} l_p \tag{9}$$

IV. Results and Discussions

A. Experimental Setup

In this paper, we used the data set provided by [16]. The data set was collected by using an Android smartphone (Huawei Mate 9 with 2.4GHz octa-core processor), equipped with a three-axis accelerometer (±8g range) and gyroscope three-axis displacement (range ±2000 degrees/sec) from Inven-Sense (ICM-20690), sampling at 100 Hz. We filtered the data to put in the train with the total number of strides is 4778 strides, the number of strides for validation is 1195, and 1494 strides for the test data. The stride length distribution in the paper is shown in Fig. 6:

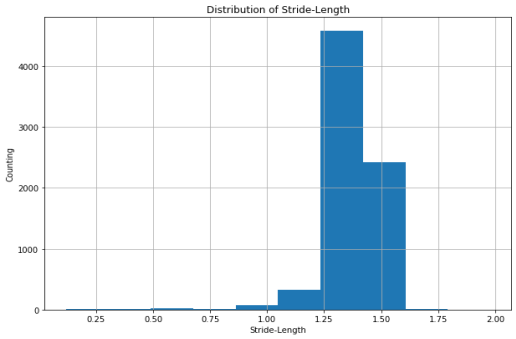

Fig. 6. Distribution of real stride length.

The data of strides with stride lengths smaller than 2 meters were used during training according to [14]. The mean of stride lengths was 1.36m.

B. Step Length Estimation Evaluation

To evaluate the effectiveness of the proposed algorithm we used the mean absolute error (MAE) according to formula (10), the relative error rate of the estimated stride length is calculated using (11),

$$MAE = \frac{1}{s}\sum_{s=1}^{S}\left|SL_e^s - SL_t^s\right| \tag{10}$$

$$MER = \frac{1}{s}\sum_{s=1}^{S}\frac{\left|SL_e^s - SL_t^s\right|}{SL_t^s}\times 100\% \tag{11}$$

where SL_e^i and SL_t^i are the estimated stride length and real stride length of the s-th test stride, respectively. S is the total number of strides.

C. Comparison with other Methods

To evaluate the performance of the proposed model, we have used the same set of training data and test data to estimate the stride length and compare the accuracy between models [14], [16]. We have rebuilt the models presented in [14], [16] with the parameters as illustrated in Table II.

Table II: List of the Parameter Values for CNN and LSTM Models

Parameter	LSTM	CNN
Batch size	128	100
Hidden layers	32-16-8-1	32-64-1024-1
Activation	ReLU	ReLU
Optimizer	RMSprop	Adam
Learning rate	0.001	0.001
Epochs	500	1000
Early stopping	50	100
Loss function	MSE	RMSE

Table III: Comparisons of Stride-Length Estimation using [8],[9],[10],[14],[16], and Proposed Method

Attributes	Weinberg [8]		Kim [9]		Ladetto [10]		CNN Hannink [14]		LSTM Wang [16]		Proposed Method	
	Error	Error Rate (%)	Error	Error Rate (%)	Error	Error Rate (%)	Error	Error Rate (%)	Error	Error Rate (%)	Error	Error Rate (%)
MAE	0.063	5.220	0.067	5.782	0.068	5.820	0.057	5.010	0.050	4.010	**0.044**	**3.750**
Std	0.080	13.470	0.088	16.000	0.087	15.899	0.079	14.440	**0.068**	**10.950**	**0.068**	11.560
25%	0.021	1.540	0.022	1.617	0.023	1.710	0.019	1.350	0.016	1.190	**0.014**	**1.010**
50%	0.044	3.210	0.047	3.424	0.048	3.440	0.040	2.870	0.032	2.330	**0.030**	**2.140**
75%	0.079	5.700	0.083	5.900	0.085	5.967	0.070	5.050	0.057	4.050	**0.049**	**3.630**
Min	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
Max	1.150	304.00	1.076	369.00	1.066	365.88	1.044	358.21	**0.837**	268.62	0.925	**244.44**

Table III shows the stride-length estimation results between the proposed method and other methods. The stride-length error and error rate results obtained by the proposed method were 0.044 m and 3.75%, respectively,

which are the best results in comparison with LSTM, CNN, Ladetto, Kim, Weinberg. As shown in Table III, the LSTM model was the 2nd best with stride-length error and error rate results were 0.050 m and 4.01%, respectively. The experimental results also demonstrated that stacked ensemble model is able to produce better SLE estimation results in the 1st, 2nd, and especially 3rd quartiles compared with other models.

Fig. 7 illustrates the performance in the testing phase of the proposed model comparing with other methods. As can be seen, the proposed stack ensemble model always delivered the same SLE accuracy regardless of training turns. On the other hand, the SLE results of the LSTM and CNN algorithms varies a lot between training turns since the estimated model parameter results of those approaches depend heavily on random initialization. Furthermore, the displayed results show that the average error of the proposed algorithms is more accurate than other methods.

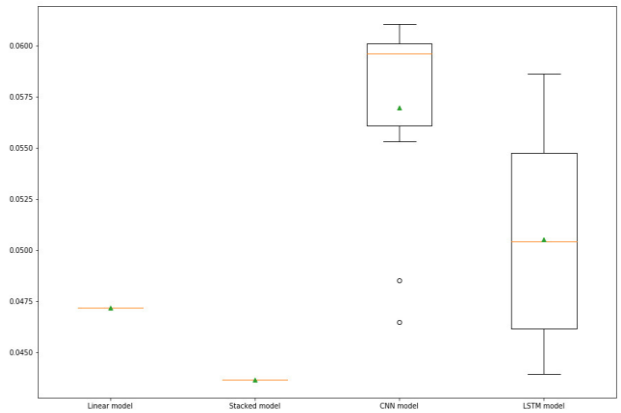

Fig. 7. Investigate the stability of the SLE algorithm.

D. Complexity

TABLE IV: THE NUMBER OF PARAMETERS OF METHODS

Model	Training dataset size	Test dataset size	Trainable Parameters	Training Time	Test Time
CNN [14]	4778 strides	1494 strides	3249889	3m29s (with GPU)	500ms
LSTM [16]			39713	27m39s (with GPU)	5s
Stacked ensemble model			61	2.92s	752ms

Table IV presents the complexity of all methods in term of execution time. The number of parameters of our method is much smaller than the other methods which use deep learning. Table IV shows the training and test time parameters for the mentioned models in this paper. The number of parameters required for training varies widely between models. In the CNN model, the number of training parameters is up to more than 3.2 million parameters while LSTM's number is nearly 40 thousand parameters. With a high-configuration computer (with GPU processor), the training time for CNN (3m29s) is much shorter compared to LSTM (27m39s), the test time is 500ms compared to 5s. The proposed method achieves very short training time (2.92s) because only the Linear Regression model needs to be trained while k-NN does not. In term of execution time for testing phase, the proposed approach is still much faster than LSTM but slightly slower than CNN approach.

V. CONCLUSION

In this paper, a stack ensemble model for stride length estimation has been proposed. The effectiveness of the proposed approach has been verified through the following aspects: the stride length estimation accuracy and the computation time. Experimental results demonstrate that the proposed approach outperforms the other state-of-the-art methods in term of stride length estimation accuracy while maintaining an acceptable computation costs. It is noted that the training time of the proposed method is much faster than the others. The execution time of the proposal proved that it is possible to utilize the proposed method for real-time indoor positioning application.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

All authors conducted the research; simulated and analyzed the data; wrote the paper; and all authors have approved the final version.

ACKNOWLEDGEMENT

This research is supported by Hanoi University of Industry (HaUI) [grant number 52-2021-RD/HĐ-ĐHCN].

REFERENCES

[1] P. Bahl and V. N. Padmanabhan, "RADAR: An In-Building RF based user location and tracking system," in *Proc. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, 2000, pp. 775-784.

[2] Y. Moustafa and A. Ashok, "The horus WLAN location determination system," in *Proc. International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2005.

[3] W. Chenshu, Y. Zheng, and L. Yunhao, "Smartphones based crowdsourcing for indoor localization," *IEEE Transactions on Mobile Computing*, vol. 14, no. 2, pp. 444-457, 2015.

[4] A. Moreira and F. Meneses, "Where@UM – Dependable organic radio maps," in *Proc. International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2015, pp. 1-9.

[5] R. Harle, "A survey of indoor inertial positioning systems for pedestrians," *IEEE Communications Surveys & Tutorials*, 15(3), 1281–1293, 2013.
[6] P. Davidson and R. Piche, "A survey of selected indoor positioning methods for smartphones," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1347–1370, 2017.
[7] L. E. Diez, A. Bahillo, J. Otegui, and T. Otim, "Step length estimation methods based on inertial sensors: A review," *IEEE Sensors Journal*, vol. 18, no. 17, pp. 6908–6926, 2018.
[8] H. Weinberg. Using the ADXL202 in Pedometer and Personal Navigation Applications. [Online]. Available: http://www.bdtic.com/DownLoad/ADI/AN-602.pdf
[9] J. W. Kim, H. J. Jang, D. H. Hwang, and C. Park, "A step, stride and heading determination for the pedestrian navigation system," *Journal of Global Positioning Systems*, vol. 3, no. 1-2, pp. 273–279, 2004.
[10] Q. Ladetto, "On foot navigation: Continuous step calibration using both complementary recursive prediction and adaptive Kalman filtering," in *Proc. 13th International Technical Meeting of the Satellite Division of The Institute of Navigation*, Salt Lake City, UT, USA, September 2000, vol. 2000, pp. 1735–1740.
[11] J. Scarlett, "Enhancing the performance of pedometers using a single accelerometer," *Analog Dialogue*, vol. 41, 2007.
[12] N. H. Ho, P. Truong, and G. M. Jeong, "Step-Detection and adaptive step-length estimation for pedestrian dead-reckoning at various walking speeds using a smartphone," *Sensors*, vol. 16, no. 9, p. 1423, 2016.
[13] I. Klein, Y. Solaz, and G. Ohayon, "Pedestrian dead reckoning with smartphone mode recognition," *IEEE Sensors Journal*, pp. 1–1, 2018.
[14] J. Hannink, T. Kautz, *et al.*, "Mobile stride length estimation with deep convolutional neural networks," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 2, pp. 354–362, 2018.
[15] F. Gu, K. Khoshelham, C. Yu, and J. Shang, "Accurate step length estimation for pedestrian dead reckoning localization using stacked autoencoders," *IEEE Transactions on Instrumentation and Measurement*, pp. 1–9, 2018.
[16] Q. Wang, L. Ye, H. Luo, A. Men, F. Zhao, and Y. Huang, "Pedestrian stride-length estimation based on LSTM and denoising autoencoders," *Sensors*, vol. 19, no. 4, p. 840, 2019.
[17] Q. Zhao, B. Zhang, J. Wang, W. Feng, W. Jia, and M. Sun, "Improved method of step length estimation based on inverted pendulum model," *International Journal of Distributed Sensor Networks*, vol. 13, no. 4, 2017.
[18] Y. Yao, L. Pan, W. Feng, X. Xu, X. Liang, and X. Xu, "A robust step detection and stride length estimation for pedestrian dead reckoning using a smartphone," *IEEE Sensors Journal*, 2020.
[19] V. Renaudin, M. Susi, and G. Lachapelle, "Step length estimation using handheld inertial sensors," *Sensors*, vol. 12, no. 7, pp. 8507–8525, 2012.
[20] Y. C. Han, K. I. Wong, and I. Murray, "Stride length estimation based on a single shank's gyroscope," *IEEE Sensors Letters*, vol. 3, no. 10, pp. 1–4, 2019.
[21] D. Kim, H. Ju, and C. G. Park, "Comparison of step length estimation models using inertial sensor on pelvis," in *Proc. First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics*, 2019.
[22] F. Gu, K. Khoshelham, J. Shang, F. Yu, and Z. Wei, "Robust and accurate smartphone-based step counting for indoor localization," *IEEE Sensors Journal*, vol. 17, no. 11, pp. 3453–3460, 2017.
[23] I. Klein, "Smartphone location recognition: A deep learning-based approach," *Sensors*, vol. 20, no. 1, p. 214, 2019.
[24] W. Ye, L. Men, and Z. Ou, "Pedestrian walking distance estimation based on smartphone mode recognition," *Remote Sensing*, vol. 11, no. 9, p. 1140, 2019.
[25] S. Vandermeeren, H. Bruneel, and H. Steendam, "Feature selection for machine learning based step length estimation algorithms," *Sensors*, vol. 20, no. 3, p. 778, 2020.
[26] Q. Wang, H. Luo, L. Ye, A. Men, F. Zhao, Y. Huang, and C. Ou, "Personalized stride-length estimation based on active online learning," *IEEE Internet of Things Journal*, pp. 1–1, 2020.
[27] I. Klein and O. Asraf, "StepNet—Deep learning approaches for step length estimation," *IEEE Access*, 2020.

Hang Thi Duong was born in Bac Giang, Vietnam, in 1978. She has been a lecturer at the Faculty of Electronic Engineering, Hanoi University of Industry since 2000. She received the B.S and M.S degree from VNU University of Engineering and Technology, in 2000 and in 2005, respectively. She is currently pursuing the Ph.D. degree in telecommunication engineering at VNU. Her main research interests are in indoor positioning systems, machine learning, and pattern classification, nature-inspired algorithm application.

Kha Manh Hoang was born in Bac Giang, Vietnam, in 1979. He received the B.E and M.E degree in Electronics and Telecommunications Engineering both from Hanoi University of Science and Technology, in 2002 and 2004, respectively. He obtained his Dr.-Eng. (PhD) degree in Communications Engineering from the University of Paderborn, Germany, in 2016 with specialization in parameter estimation of missing data and application to indoor positioning. He is now working as a vice dean of Faculty of Electronics, Hanoi University of

Industry. He has served as the CoTPC Chairs of RICE (2019), TPC member of ATC (2018, 2019), reviewer of REV-ECIT, ATC, Journal of Science and Technology (ISSN 1859-3585). His research interests include digital signal processing, wireless communication, positioning engineering, machine learning and pattern classification, and nature-inspired algorithm application.

Hung Viet Pham was born in Hanoi, Vietnam, in 1988. He received the B.E in Electronics and Telecommunications Engineering from the University of Transport and Communication in 2011. He obtained his M.E degree in Computer and Information Science from Hanoi University of Science and Technology in 2017. He is now working as Data Scientist at Trusting Social. His research interests include pattern recognition and machine learning.

Vu Anh Trinh was born in Hanoi, Vietnam, in 1958. He received BS (1983) in Radio physic from Hanoi University, Ph.D. in Mathematic and Physic (1994) from Moscow State University (named M.V.Lomonosov) in Russia. Exchange researcher (2002) at Tasmania University, Australia. Currently is an Associate Professor in the Department of wireless Communications at Faculty of Electronics and Telecommunication (FET), University of Engineering and Technology (UET), Vietnam National University, Hanoi (VNU Hanoi). He is a member of Radio Electronic Vietnam Associate. Interest research fields: 5G Wireless Communications, mMTC and URLL systems, Caching and Computing Systems.