JST: Smart Systems and Devices
Volume 32, Issue 3, September 2022, 001-008

An Improved Sleeping Posture Recognition Using Pressure Sensor Data and Deep Learning

Huy Hoang Nguyen*[*], *Huu Trung Nguyen, Thuy Anh Nguyen
Hanoi University of Science and Technology, Hanoi, Vietnam
[*]Corresponding author email: hoang.nguyenhuy@hust.edu.vn

Abstract

Many chronic diseases, such as cardiovascular disease and sleep disorders, can be diagnosed and treated by assessing the sleeping quality. Sleep posture recognition is an important part of determining sleep quality in sleep research. While most recent studies focus on the classification problem with the number of sleeping postures being equal to or less than ten, this study aims to achieve state-of-the-art results on 17 in-bed position classification. To do this, a spatial pyramid pooling module was added to the top of the EfficientNet B0 model and the contrastive loss and cross-entropy loss functions are combined to act as the main loss function. The random 10% salt and pepper noise are utilized to augment the training data. Experimental results confirm that the proposed approach achieves the best accuracy of 96.01% and outperforms the existing methods. Additionally, we also provide an estimation related to the impact of various combinations of backbone models and loss functions on the performance of the classifier.

Keywords: Sleeping posture classification, pressure sensor data, deep learning.

1. Introduction

In-bed posture recognition plays a vital role in sleep studies [1]. Indeed, doctors can diagnose esophagus problems earlier by monitoring a patient’s sleeping positions for a period. Many studies show that resting on your right side increases your chances of developing transitory lower esophageal sphincter relaxation, a significant cause of nocturnal gastroesophageal reflux. In the hospital environment, bed-ridden patients easily get bed-sore due to lying in the same position for a long time. To prevent pressure ulcers in bed-ridden patients, caregivers must frequently monitor and adjust the patients’ sleeping positions. This problem leads to the over workload problem for nurses and caregivers. Therefore, automatic in-bed posture recognition is helpful to detect unreasonable sleeping postures and remind clinical staff to change the patient’ sleeping position.

Using pressure sensors to recognize in-bed sleeping positions has many advantages. Unlike the radiofrequency signal-based solution, this method does not necessitate the use of two devices, one on each side of the bed. Compared to RGB camera-based solutions, this approach does not violate patients' privacy. This method also does not cause discomfort to the patients like the wearable device-based solutions. The pressure sensor-based solutions are not affected by the low light condition.

Related works of using pressure sensor data can be divided into two categories consisting of the traditional method and the deep learning method. In the traditional method, the features of sleeping postures are investigated and generated by manual extraction techniques. In the deep learning method, the features of sleeping positions are learned during the training of the Convolution Neural Network (CNN) model. Due to the advantage of feature learning, recent solutions have widely utilized various CNN models to increase the accuracy of the classifier.

Indeed, several CNN-based studies have been proposed, such as the works of [2-4]. In [2], the researchers introduced a self-supervised learning model which consists of an upstream self-supervised pre-training task and a downstream recognition task. A four-layer CNN model with rotated data was used in the upstream part to enhance the multi-dimensional feature representation learning, whereas a combination of bidirectional Long-Short Term Memory (LSTM) and conditional random field was applied in the downstream part to produce the sleeping posture labels. In [3], Yu *et al.* proposed a bunch of data augmentation techniques and a five-layer CNN model to solve the small data-size problem. Both these approaches provide impressive results with over 99% accuracy in the 3-class dataset. In the work of Costello *et al.*, the authors combined a fuzzy rule inference with a mixture of CNN and LSTM models to achieve the accuracy of 98.8% in terms of 10 sleeping postures classification [4].

To solve the problem of classifying more than 10 sleeping postures, Davoodina *et al.* proposed

ISSN: 2734-9373
https://doi.org/10.51316/jst.160.ssad.2022.32.3.1
Received: February 22, 2022; accepted: August 2, 2022

a CNN model including convolution blocks combined with Batch Normalization, Maxpool, and LeakyReLU [5]. The research showed 99.9 % and 87 % accuracy with 3 and 17 classes, respectively. This result was improved by the work of Doan *et al.* [6]. In [6], the raw pressure sensor data was normalized to the range of image data, and then the obtained data were smoothed and denoised by a preprocessing technique. To train the EfficientNet B0-based classifier, the authors applied the AM - Softmax loss function and the Adam optimizer with L2 regularization. Doan *et al.* achieved the best result at 95.32 % in the LOSO cross-validation scheme.

Considering the problem of the high-power consumption of the CNN model, Dam *et al.* [1] proposed a Spiking Neural Network (SNN) model to classify 17 sleeping postures. In their work, a modification of Darknet19 model was trained, and then the best pre-trained model was converted to the SNN model with several pre-defined rules. Their experimental results point out that the SNN model provides better power consumption performance than CNN. However, their accuracy is lower than the work of Doan *et al.* about 4.7%.

It can be seen from the above analysis that none of the studies evaluates the impact of various combinations of backbones and loss functions on the performance of sleeping posture classification. Additionally, the results on the 17-class dataset can be still improved. Therefore, we focus on achieving new state-of-the-art results by searching for a new combination of the backbone network and loss function. Since none of the previous studies has attempted this work, our approach provides three main research contributions. The first contribution is to estimate the performance of various backbone networks and loss functions in terms of sleeping position classification. The second contribution is a modification of EfficientNet B0 combined with supervised contrastive learning to achieve a state-of-the-art result for 17 in-bed postures classification. The final contribution is a novel in-bed posture classification algorithm.

The rest of this study is designed as follows: Section 2 details our proposed approach to sleeping posture classification; Section 3 outlines and discusses our experimental results; Section 4 provides a conclusion and a suggestion for future work in this field.

2. Methodology

2.1. Dataset

In this work, we utilized the Pmatdata dataset [5] to evaluate our proposed algorithm. This dataset was obtained from a pressure sensor map named Vista Medical FSA SoftFlex which is designed by 2048 sensors and organized with a resolution of 32 x 64. The process of data acquisition was implemented at the sampling rate of 1Hz from 13 individuals (S1-S13) with various heights and weights in 17 lying positions. Table 1 presents the details of 17 classes in the Pmatdata dataset.

Table 1. 17 classes of the Pmatdata dataset

Class	Icon	Name	Class	Icon	Name
1		Supine	10		Supine Knees up
2		Right	11		Supine Right Knee up
3		Left	12		Supine Left Knee up
4		Right 30° Body-roll	13		Right Fetus
5		Right 60° Body-roll	14		Left Fetus
6		Left 30° Body-roll	15		Supine 30° Bed Inclination
7		Left 60° Body-roll	16		Supine 45° Bed Inclination
8		Supine star	17		Supine 60° Bed Inclination
9		Supine Hand Crossed			

2.2. Preprocessing Technique

To enhance the accuracy of the proposed classification method, we implemented the same image preprocessing technique that was employed in the work of Doan *et al.* [6] before feeding the pressure data image into the classifier. Since the range of raw data is from 0 to 10000, we firstly normalized the raw data to the range of the 1-channel image. Subsequently, we slid a 3x3 kernel to transform each serial 3 data frame into a 3-channel image. To denoise the obtained 3-channel image, we applied a spatio-temporal 3x3x3 median filter. Finally, we utilized an equalized histogram algorithm to enhance the contrast of the filtered image. Fig. 1 shows 9 samples of the pressure sensor system before and after applying the proposed image preprocessing technique.

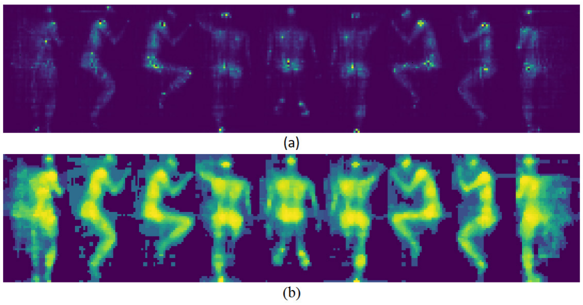

(a)
(b)

Fig. 1. Samples before and after applying the proposed image preprocessing technique

2.3. An Improved Sleeping Posture Classification Model

The preprocessed pressure image was fed into a deep model for feature extraction and classification. We utilized the EfficientNet B0 as our backbone network for the classifier. The advantage of Spatial Pyramid Pooling (SPP) was proven in many studies [7-9]. Therefore, we added the SPP module to the top of the EfficientNet B0 to enhance the capability of feature extraction and learning. Instead of using the traditional Softmax loss function, a combination of the contrastive loss function and cross-entropy loss function was adopted during the training. Additionally, the training data was augmented by adding the random 10% salt and pepper noise.

2.3.1. Network architecture

Based on the original EfficientNet B0 model, several modifications have been created to leverage accuracy. Firstly, the input size of the network model is set to 112 x 224, which is the same ratio as the preprocessed pressure image. Secondly, a convolution layer with 256 1x1 filters rests on top of the EfficientNet B0 model. After the convolution layer, a SPP layer is added to enhance the capability of feature extraction and learning of the proposed model. The SPP layer pools the features and generates fixed-length outputs, which are then fed into the next fully connected layer [7]. In our work, the SPP layer output is aggregated from outputs of four max-pooling layers whose filter sizes are 4x7, 2x4, 2x3 and 1x2. Table 2 presents our SPP-EfficientNet B0-based classification model.

Table 2. Details of our SPP-EfficientNet B0-based classification model

Layer	Resolution	Channels	Layers
Conv3x3	112 x 224	32	1
MBConv1, k3x3	56 x 112	16	1
MBConv6, k3x3	56 x 112	24	2
MBConv6, k5x5	28 x 56	40	2
MBConv6, k3x3	14 x 28	80	3
MBConv6, k5x5	7 x 14	112	3
MBConv6, k5x5	7 x 14	192	4
MBConv6, k3x3	3 x 7	320	1
Conv1x1	3 x 7	1280	1
Adaptive Average Pooling	3 x 7	1280	1
Conv2D	3 x 7	256	1
SPP	1 x 1	3584	1
Linear	1 x 1	17	1

2.3.2. Proposed loss function

Cross-entropy loss function

In the task of classification, we utilized the cross-entropy loss function which is one of the most common categorical losses that has been used over the years. The cross-entropy loss is defined in this formula:

$$L_{CE} = -\Sigma_{i=1}^{C} t_i \log(\frac{e^{s_i}}{\Sigma_{j=1}^{C} e^{s_j}}) \qquad (1)$$

where *C* represents the total number of classes, t_i and s_i are the ground truth and the CN score for class i^{th} , respectively.

Contrastive loss function

Unlike the cross-entropy loss function, which is supposed to learn to directly predict the label, a value, or a set of values given by input, the objective of the contrastive loss function is to guide the samples from the same class to be mapped to the nearby features. The CNN model can be considered as a mapping function that transforms input samples to the output manifold. The contrastive loss function [10] is expected to help the model get better-embedded representations in the feature extraction task. In the output space, simple distance metrics (such as Euclidean distance) should replicate neighborhood relationships in the input space.

Consider set I of training samples X_i, for each $X_i \in I$, there is a set S_{X_i} of training samples that have the same ground truth as X_i. Let $X_1, X_2 \in I$ be a pair of input samples. Let Y be the binary label assigned to this pair, $Y = 0$ if X_1, X_2 have same ground truth label and $Y = 1$ if they are deemed dissimilar. Define the distance D_w between X_1, X_2 as the Euclidian distance between the outputs of the model:

$$D_w(X_1, X_2) = ||G_w(X_1) - G_w(X_2)||_2 \qquad (2)$$

where G_w represents the CNN model which is parameterized by the weight W.

The contrastive loss function is represented in this formula:

$$L_{Con}(W) = \Sigma_{i=1}^{P} L(W, (Y, X_1, X_2)^i) \qquad (3)$$

where P represents the number of training pairs and $(Y, X_1, X_2)^i$ is the i^{th} labeled sample pair. In the formula, $L(W, (Y, X_1, X_2)^i)$ is the loss calculated on the i^{th} sample pair. The detailed definition of the loss on a training pair is:

$$L(W, Y, X_1, X_2) = \frac{1}{2}(1 - Y)\, D_w(X_1, X_2)^2 + \frac{1}{2} Y\{\max(0, m - D_w(X_1, X_2)\}^2 \qquad (4)$$

where $m > 0$ is a margin. When the representation produced for a dissimilar pair is at a sufficient distance, no efforts are wasted on enlarging the distance. This means that further training can focus on the more difficult pairs.

A combination of cross-entropy and contrastive loss functions

To take advantage of both the cross-entropy and contrastive loss function, we propose the mixture loss with the coherency of both losses defined in this formula:

$$L_{total} = \alpha \times L_{CE} + \beta \times L_{Con} \tag{5}$$

where α and β are the coefficients of cross-entropy loss and contrastive loss, respectively. For each input sample X_1, the cross-entropy loss is calculated with the corresponding target while X_1 and another random sample X_2 are fed to compute the contrastive loss.

2.3.3. Training algorithm

In the work of training the model with the loss we proposed, we implemented a learning architecture called *siamese* architecture [10]. This was made up of two copies of our SPP-EfficientNet B0 model which shared the same set of parameters W in (2). The outputs of this architecture were fed to our proposed loss module to compute the combination loss. We determined that the input to the entire system was a pair of images (X_1, X_2) and a label Y of the pair. To warrant the balance in training pairs, we firstly took image X_1 and label Y before randomly choosing X_2 based on Y, X_1. To begin with, the training sample X_1 was fed into the first copy of our model in *siamese* architecture which returned the predicted logit $G_w(X_1)$ for classification. The predicted logit $G_w(X_1)$ and the corresponding target of X_1 were used to compute the cross-entropy loss function. After that, we let image X_2 pass through the second copy in the system to get the output $G_w(X_2)$. The contrastive loss combined the distance $D\big(G_w(X_1), G_w(X_2)\big)$ with label Y to produce the scalarloss. Finally, the combination loss was figured out and based on that, the parameters W were updated.

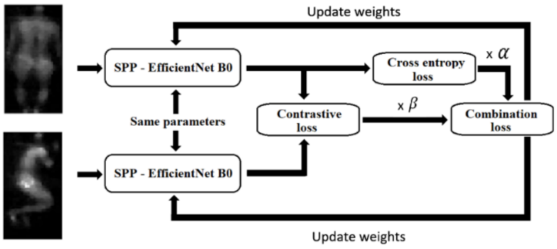


Fig. 2. Our proposed training algorithm

To decrease the training time, the pretrained EfficientNet B0 was utilized during the network training. We employed an Adam optimizer in the training phase with a learning rate of 0.005 and a step learning rate scheduler with a decay rate of 0.95 for every 10 epochs. We trained the network in 30 epochs with a batch size of 32. We used L2-regularization with a weight decay coefficient of 0.002 to avoid the over-fitting problem. Additionally, the random 10% salt and pepper noise were added to augment the training data. Fig. 2 presents our proposed training procedure.

3. Experimental Results

3.1. Experimental Setups

For the training stage, our SPP-EffcientNet B0-based classification model was trained on a computer with an Intel Core E5 Xeon - 2650 V4 CPU @ 2.2 GHz x 12 and 64 GB RAM running on a 64-bit Ubuntu 18.04 operating system. Four RTX2080Ti GPU were utilized to accelerate the training process. For validating our proposed method, we trained our model based on two validation schemes, namely, *k*-fold and LOSO. In the *k*-fold cross-validation, the dataset was divided into *k* folds in which one fold was used for testing, and the rest were used for training in one validation round. The validation round was repeated for every fold. The *k*-fold accuracy was the average score of all validation rounds. In the LOSO cross-validation, in each validation round, one subject was kept aside for testing, and the other subjects were utilized for training. We repeated the training 10 times. After calculating the accuracy of all the testing sets, the LOSO accuracy was the average value of all accuracy scores.

3.2. Training Results

Four experiments were implemented in the training stage. The first experiment combined various backbone networks and loss functions to estimate the impact of these combinations on the classification performance. The second experiment was the training process of the proposed classification to find the best model. The next experiment was an ablation study of various combinations of SPP, EfficientNet B0 model, salt and pepper noise-based data augmentation and loss functions. The final experiment is to visualize the deep features of the proposed model with three loss functions (Cross-Entropy, Contrastive and a combination of Cross-Entropy and Contrastive) to compare their capability in terms of classification.

3.2.1. Experiment 1

In this experiment, various backbone models and loss functions were combined to train the classifier for the task of 17 sleeping posture classification. The chosen backbone networks were Resnet50, Inception V3, Mobilenet V2 and EfficientNet B0, while the chosen loss functions were Softmax (S), L-Softmax (L-S), A-Softmax (A-S), AM-Softmax (AM-S). Additionally, several other loss functions were mixed with the Cross-Entropy loss (CE) function: Circle loss (Cir), Contrastive loss (Con), Center loss (Cen), Triplet loss (Trip) and Triplet Center loss (Trip-Cen). Details of the training procedure including the optimizer, learning rate, step learning rate scheduler, decay rate, and batch size were the same as those employed in the work of Doan *et al.* [6].

Table 3 represents how well the different loss functions can be combined with the CE loss to help to enhance the performance of various CNN models. In general, there are no considerable points in 10-fold cross-validation when the used models and losses are impressively robust. However, the mixed loss has a remarkable impact on the model's classification ability when looking into LOSO Cross-Validation. Among various mixed loss functions, the combination of Con loss and CE loss brings outstanding outcomes in various models. Specifically, EfficientNet B0 and ResNet50 are the best, achieving 95.41% and 95.62%, respectively. Therefore, this experiment concludes that the mixture of Con loss and CE loss is the most suitable case for the task of pressure sensor data based on 17 sleeping postures classification.

Even though this result shows that the combination of the CE-Con loss function and the Resnet50 provides higher accuracy than that of the nevertheless utilized the EfficientNet B0 to develop a new backbone model for this study due to its obvious advantage compared to the Resnet50. According to [11], the total parameters and BFlops of the EfficientNet B0 are lower by 4.9 times and 11 times, respectively, than those of the Resnet50. Additionally, the EfficientNet B0 model size is only 15 MB, whereas the Resnet50 model size is 85 MB. Therefore, EfficientNet B0 requires less memory usage than Resnet50. As a result, the EfficientNet B0 is more suitable than Resnet50 for implementing the in-bed posture classification task on edge devices such as Jetson Nano, TX2 or Xavier.

Table 3. Average accuracy of CNN models with various loss functions on two cross-validation schemes

Loss functions	**Cross-Validation Schemes**							
	Resnet50		**InceptionNet-V3**		**MobileNet-V2**		**EfficientNet B0**	
	10-fold	**LOSO**	**10-fold**	**LOSO**	**10-fold**	**LOSO**	**10-fold**	**LOSO**
S	99.99	93.92	99.99	91.84	99.99	94.26	99.98	93.56
L-S	99.97	92.67	99.95	92.57	99.97	91.60	99.99	92.52
A-S	99.99	93.73	99.96	92.20	99.99	93.43	99.97	94.34
AM-S	99.99	93.89	99.99	93.37	99.98	94.62	99.99	95.32
CE-Cir	100	94.59	99.94	93.92	100	94.11	99.95	93.38
CE-Cen	99.86	91.21	99.95	94.5	99.93	93.39	99.96	95.3
CE-Trip	99.73	94.35	99.65	93.82	100	93.09	98.89	94.84
CE-Trip-Cen	100	92.29	99.99	94.35	99.99	93.36	99.98	93.12
CE-Con	99.98	**95.62**	99.98	**94.63**	99.96	**94.77**	99.98	**95.41**

Table 4. Average accuracy of the proposed method with various parameter pairs (α,β)

Parameters (s, m)	**Accuracy on the LOSO validation dataset (%)**													
	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**	**S10**	**S11**	**S12**	**S13**	**Average**
(α = 1 , β = 1)	100	100	99.7	86.8	87.9	86.7	91.3	100	93.8	99.3	100	99.7	94	95.32
(α = 1 , β = 0.9)	97.8	99.2	99.5	85.9	86.3	91	94.2	100	93.7	99.8	99.9	99.9	93.9	95.46
(α = 1 , β = 0.8)	96.1	96	99.6	84.2	87.2	91.3	92.3	100	93.7	99.3	99.6	99.8	93.6	94.82
(α = 1 , β = 0.7)	96.4	99.8	99.9	87.5	87.3	87.7	92.9	100	96	98.1	100	99.9	93	95.26
(α = 1 , β = 0.6)	99.8	99.6	99.9	83.6	87.6	88.1	93.2	100	94.3	99.5	98.7	99.2	94.2	95.2
(α = 1 , β = 0.5)	99.9	100	100	86.8	87.5	90.1	94.8	100	94.5	99.4	99.7	99.9	94.7	95.94
(α = 1 , β = 0.4)	99.6	99.5	99.9	81.1	81	86.8	95.2	100	94.4	97.8	99.9	99.1	93.3	94.43
(α = 1 , β = 0.3)	96.3	97.5	98.4	82.1	87.5	90	92.3	100	93.9	99.7	99.7	99.9	94.1	94.72
(α = 1 , β = 0.2)	97.9	98.1	99.8	84.1	83.9	90.8	91.2	100	93.9	99.2	99.9	99.8	91.8	94.64
(α = 1 , β = 0.1)	97.3	98.4	99.8	82.8	85	86.7	93.2	100	93.3	100	99.7	99.8	93	94.53
(α = 0.1 , β = 1)	96.9	99.5	99.8	81.5	83.8	91.7	93.3	100	94.7	99.9	100	99.9	94.3	95.02
(α = 0.2 , β = 1)	97.6	95.7	100	84.2	86.1	89.8	95.9	100	94.2	99.8	99.6	99.7	96	95.27
(α = 0.3 , β = 1)	97.7	99.2	98.8	83.5	84	92.8	96.7	100	96.3	99.3	99.3	100	94.4	95.53
(α = 0.4 , β = 1)	**99.6**	**100**	**100**	**86.9**	**86.5**	**89.3**	**96.5**	**100**	**95.2**	**99.8**	**100**	**100**	**94.4**	**96.01**
(α = 0.5 , β = 1)	96.7	99.6	100	81.1	96.9	87	94.1	100	94.2	97.8	100	100	94.3	95.51
(α = 0.6 , β = 1)	99.8	99.7	100	82.2	85	86.8	94.3	100	93.4	98.9	100	100	93.1	94.86
(α = 0.7 , β = 1)	98.9	99.9	100	81	87.2	88	93.8	100	92.9	99.1	99.9	99.9	93.5	94.93
(α = 0.8 , β = 1)	99.1	99.4	99.7	82.1	87.2	88.4	93.5	100	93.2	99.7	99.8	100	94.1	95.09
(α = 0.9 , β = 1)	99.2	99.5	99.8	81.8	83.8	87.3	94.2	100	94	99.1	100	99.9	91.3	94.6

Table 5. Ablation study on combinations of EfficientNet model, data augmentation and loss function

Training cases	**Average accuracy on two cross-validation schemes**		
	5-fold	**10-fold**	**LOSO**
EfficientNet B0 + CE loss function	99.90	99.91	93.29
EfficientNet B0 + Con loss function	99.88	99.89	94.51
EfficientNet B0 + CE-Con loss function	99.92	99.89	95.41
EfficientNet B0 + CE loss function + data augmentation	99.93	99.92	93.72
EfficientNet B0 + Con loss function + data augmentation	99.92	99.91	94.67
EfficientNet B0 + CE-Con loss function + data augmentation	99.95	99.97	95.74
SPP-EfficientNet B0 + CE loss function	99.89	99.91	93.63
SPP-EfficientNet B0 + Con loss function	99.94	99.93	94.69
SPP-EfficientNet B0 + CE-Con loss function	99.95	99.97	95.69
SPP-EfficientNet B0 + CE loss function + data augmentation	99.92	99.93	93.67
SPP-EfficientNet B0 + Con loss function + data augmentation	99.94	99.97	95.53
SPP-EfficientNet B0 + CE-Con loss function + data augmentation	**99.99**	**99.99**	**96.01**

3.2.2. Experiment 2

To optimize the performance of the proposed SPP-EfficientNet B0 model, we tested various parameter pairs (α,β) of the CE-Con loss function and observed how they affected the average accuracy in LOSO cross-validation. The statistical outcomes of each case are shown in Table 4, which can be divided into 2 sub-tables. We keep unchanged α equal to 1 and range the β value from 0.1 to 1 and do the same with otherwise; 19 cases were implemented for this experiment. With various pairs of parameters, the downside of the model is associated with S4, S5, and S6, which are usually lower than 90%. In contrast, (S1, S2, S3, S8, S10, S11, and S12) significantly contribute to the results when the accuracy is around 99%. More importantly, the adjustment of the mentioned parameters increases the model's average accuracy by 1.6% and achieves the best model when it turns to the pair of α = 0.4 and β = 1. As a result, this pair of parameters is used in the proposed solution.

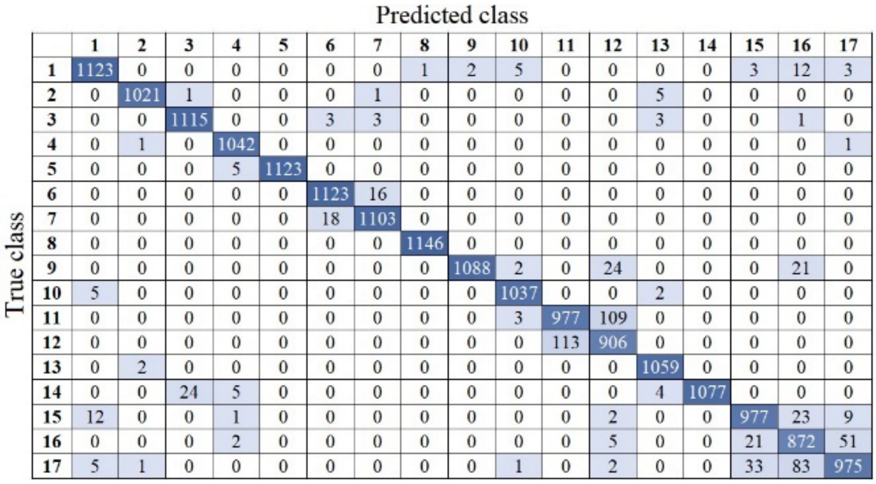
3.2.3. Experiment 3

In this experiment, we conducted an ablation study to verify the advantage of the proposed SPP block and salt-pepper noise-based data augmentation. The parameter was fixed for the proposed combination loss function. The ablation experiment results with two cross-validation schemes are reported in Table 5. This table shows the comparison results of various combinations of SPP, EfficientNet B0 model, data augmentation and loss function. It can be observed that the proposed approach provides the best performance in both 5-fold and 10-fold cross-validation. Compared to the traditional combination of EfficientNet B0 and CE loss function, the appearance of the SPP block, data augmentation and CE-Con loss function resulted in an improvement (from 93.29% to 96.01%) in the LOSO cross-validation scheme. These ablation experiment results also indicate that the proposed combination is better than others. The confusion matrix of the proposed solution is shown in Table 6.

3.2.4. Experiment 4

To more deeply analyze the advantage of the CE-Con loss function compared to the CE loss function and Contrastive loss function, we visualized the deep feature of the last layer of our proposed model during the training procedure. Fig. 3 shows the distribution of learned features under three loss supervisions. In Fig. 3 (a), the points with different colors correspond to features from various classes. We can see that the deeply learned features are separable in angle direction under the supervision of this loss. However, the deep features are not discriminative enough since they still contain significant intra-class variations. Accordingly, it is not appropriate to directly use these features for recognition. In Fig. 3 (b), we observe that learned features in the same class are clustered in a low-variance region. As a result, Contrastive loss enhances the discriminative power of the learned features. It increases the distance between classes and in turn, narrows the gap between the same classes. However, the inter-class separability is not sufficient. Fig. 3 (c) shows that the combination of contrastive loss and cross-entropy loss performs better clustering than the previous methods. This method simultaneously achieves good intra-class compactness and inter-class separability.

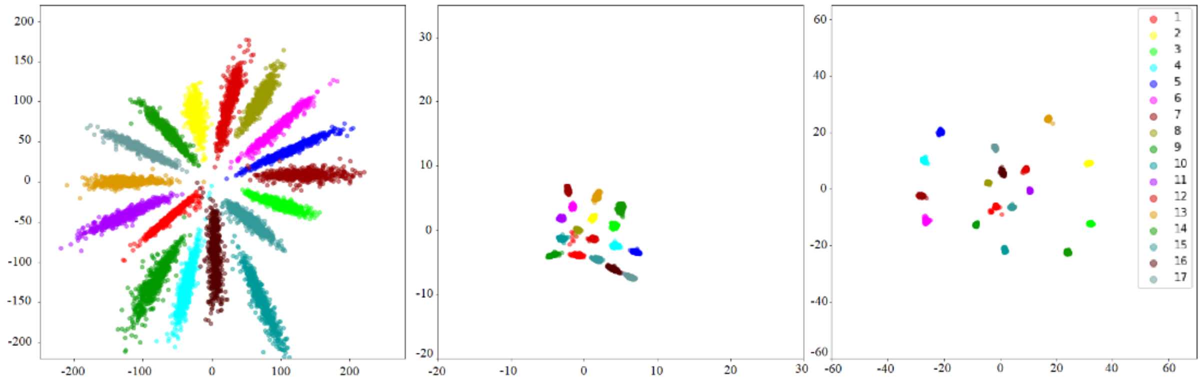
Table 6. The confusion matrix of the best model



Predicted class

True class

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	1123	0	0	0	0	0	0	1	2	5	0	0	0	0	3	12	3
2	0	1021	1	0	0	0	1	0	0	0	0	0	5	0	0	0	0
3	0	0	1115	0	0	3	3	0	0	0	0	0	3	0	0	1	0
4	0	1	0	1042	0	0	0	0	0	0	0	0	0	0	0	0	1
5	0	0	0	5	1123	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1123	16	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	18	1103	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	1146	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1088	2	0	24	0	0	0	21	0
10	5	0	0	0	0	0	0	0	0	1037	0	0	2	0	0	0	0
11	0	0	0	0	0	0	0	0	0	3	977	109	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	113	906	0	0	0	0	0
13	0	2	0	0	0	0	0	0	0	0	0	0	1059	0	0	0	0
14	0	0	24	5	0	0	0	0	0	0	0	0	4	1077	0	0	0
15	12	0	0	1	0	0	0	0	0	0	0	2	0	0	977	23	9
16	0	0	0	2	0	0	0	0	0	0	0	5	0	0	21	872	51
17	5	1	0	0	0	0	0	0	0	1	0	2	0	0	33	83	975



(a) Cross Entropy loss

(b) Contrastive loss

(c) Our combination loss

Fig. 3. The distribution of learned features under three loss supervisions

Table 7. Comparison of inference time on various Jetson devices

Method	Average inference time on various Jetson devices (ms)		
	Nano	**TX2**	**Xavier**
Doan et al. [6], 2021	227	136	65
Our proposed method	230.1	137.4	65.6

3.3. Inference Time on Various Devices

Table 7 reports the inference time of the proposed method on various devices. Although the training stage costs highly processing time due to the large computation of CE-Con loss function, the inference time of the proposed model still satisfies the real-time requirement for various pressure sensor systems. The processing time of our solution is only 3.1 ms, 1.4 ms and 0.6 ms higher than that of Doan *et al.* on Nano, TX2 and Xavier, respectively. In the inference stage, the CE-Con loss computation is removed; hence this timing difference is caused by the addition of the SPP module. According to Doan *et al.* [6], their solution satisfies the real-time requirement with the data sampling rate of 0.3 Hz, 1 Hz, 2 Hz and 4 Hz. Therefore, our proposed solution also passes the real-time requirement at these data sampling frequencies.

3.4. Comparison with Other Methods

In this experiment, we compare the obtained results to related works in the same sensor mattress resolution. It can be observed from Table 8 that the CNN-based solution group not only provides better performance but also possesses the ability to classify more classes than the binary pattern matching-based method. In the experiments with the Pmatdata dataset, the research of Davoodnia *et al.* achieved 93.2% accuracy in 10-fold cross-validation and only 87% in LOSO validation, whereas the recent study of Doan *et al.* achieved outstanding classification with 99.9% and 95.32%, respectively. Our proposed method is an improvement on Doan *et al.*'s research being 0.69% higher in LOSO validation and near-perfect in its performance of the 5-fold and 10-fold cross-validation schemes. Compared to the work of Dam *et al.,* our approach provides better performance. These results confirm that our work achieves a state-of-the-art result in terms of 17 in-bed posture classification.

Table 8. A comparison with other methods

Studies	**Database**	**Algorithm**	**Postures**	**Average accuracy**		
Author name, year	**(Subjects)**			**5-fold**	**10-fold**	**LOSO**
Pouyan et al. [12], 2013	20	Binary pattern matching	8	-	97.1	-
Davoodnia et al. [5], 2019	13	Multi-tasking CNN model + Softmax loss function	17	-	93.2	87
Doan et al. [6], 2021	13	EfficientNet B0 model + AM-Softmax loss function	17	99.98	99.97	95.32
Dam et al. [1], 2021	13	Spiking Neural Network	17	99.95	99.96	90.56
Our proposed method	**13**	**SPP-EfficientNet B0 model + Combination loss function**	**17**	**99.99**	**99.99**	**96.01**

4. Conclusion

This paper presented a novel approach to improve the sleeping posture classification based on pressure sensor data. The proposed method is a combination of a preprocessing technique and an SPP-EfficientNet B0-based classifier with a combined cross-entropy and contrastive loss function. With 99.9% accuracy for *k*-fold cross-validation and 96.01% for LOSO cross-validation, our method achieves a new state-of-the-art result for the classification of 17 postures on the Pmatdata dataset. The experimental results also reveal that the utilization of a combination loss can enhance the accuracy of the classifier. In terms of future work, we focus on developing the Deep Spiking Neural Network to reduce the power consumption of the CNN model and maintain the high accuracy of this work's results.

Acknowledgments

This research is funded by Hanoi University of Science and Technology under grant number T2021 - PC - 012.

References

[1] H. P. Dam, N. D. A. Pham, H. M. Pham, N. P. Doan, D. M. Nguyen and H. H. Nguyen, In-bed posture classification using pressure sensor data and spiking neural network, in Proc. NICS, 2021, pp. 358-363, https://doi.org/ 10.1109/NICS54270.2021.9701

[2] A. Zhao, J. Dong and H. Zhou, Self-supervised learning from multi-sensor data for sleep recognition, IEEE Access, vol. 8, pp. 93907-93921, 2020, https://doi.org/ 10.1109/ACCESS.2020.2994593.

[3] E. Yu, M. Kenji, Data augmentation to build high performance dnn for in-bed posture classification, Journal of Information Processing, vol. 26, pp. 718-727, 2018, https://doi.org/10.2197/ipsjjip.26.718.

[4] F. J. Costello *et al.,* Exploring a fuzzy rule inferred convlstm for discovering and adjusting the optimal posture of patients with a smart medical bed, Int. J. Envi & Res. Pub Health, 2021, vol. 18, no. 12, pp. 1-14, https://doi.org/10.3390/ijerph1812634.

[5] V. Davoodnia and A. Etemad, Identity and posture recognition in smart beds with deep multitask learning, in Proc. SMC, 2019, pp. 3054-3059, https://doi.org/ 10.1109/SMC.2019.8914459.

[6] N. P. Doan, N. D. A. Pham, H. M. Pham, H. T. Nguyen, T. A. Nguyen and H. H. Nguyen, Real-time sleeping posture recognition for smart hospital beds, in Proc. MAPR, 2021, pp. 1-6, https://doi.org/ 10.1109/MAPR53640.2021.9585289

[7] K. He, X. Zhang, S. Ren and J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, in Proc. ECCV, 2015, pp. 346-361, https://doi.org/10.1007/978-3-319-10578-9_23

[8] H. H. Nguyen, T. N. Ta, N. C. Nguyen, V. T. Bui, H. M. Pham and D. M. Nguyen, YOLO based real-time human detection for smart video surveillance at the edge, in Proc. ICCE, 2021, pp. 439-444, https://doi.org/ 10.1109/ICCE48956.2021.9352144

[9] H. P. Nguyen, T. P. Hoang and H. H. Nguyen, A deep learning based fracture detection in arm bone X-ray images, in Proc. MAPR, pp. 1-6, https://doi.org/ 10.1109/MAPR53640.2021.9585292.

[10] R. Hadsell, S. Chopra and Y. LeCun, Dimensionality reduction by learning an invariant mapping, in Proc. CVPR, 2006, pp. 1735-1742, https://doi.org/ 10.1109/CVPR.2006.100.

[11] M. Tan and Q. V. Le, EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, 2019. [Online]. Available: https://arxiv.org/abs/1905.11946

[12] M. B. Pouyan, S. Ostadabbas, M. Farshbaf, R. Yousefi, M. Nourani and M. D. M. Pompeo, Continuous eight-posture classification for bed-bound patients, in Proc. BEI, 2013, pp. 121-126, https://doi.org/ 10.1109/BMEI.2013.67469.