Identity and Posture Recognition from Smart Bed Pressure Data with Inception Based Deep Network

Michele M. Crudele[†], Filippo Ziliotto[‡]

***Abstract*—Together with other physiological activities, sleep has been shown to have a significant impact on different aspects of human health. Monitoring sleep posture can give valuable information not only to improve sleep quality, but also to prevent diseases such as pressure ulcers or sleep apnea. In this study, we outperform the state of the art model in the classification of both subjects and 17 different in-bed postures, using a dataset made up of pressure maps from 13 participants, collected with a smart bed. The model we propose is built from scratch after a study about three standard deep learning architectures, implementing a simplified version of the *GoogleNet*, using inception blocks. An improvement of about 10% is achieved in the accuracy of both tasks when testing the model with augmented data and validating it using K-Fold and LOSO (Leave One Subject Out) schemes. This means that our model is more accurate either with imperfect and new subjects' pressure maps, achieving almost 100% accuracy also when testing the model with augmented data in the k-fold validation scheme. As a consequence, it would be more suitable to be embedded in clinical or smart home applications for sleep posture monitoring.**

***Index Terms*—Sleep Posture Monitoring, Smart Bed, Pressure Maps, CNN Automated Feature Extraction, Multi-Output Classification, Inception Block.**

I. Introduction

[1] In the last few years an always higher interest in human physiological activities has grown, leading to the introduction of many different devices with embedded sensors to monitor them. Sleep is one of those activities, found to be linked to various aspects of human physiology. In particular, studies about in-bed posture have proved its key role in sleep quality and its high influence on sleep diseases such as apnea [1] or pressure ulcers [2]. That is why sleep posture recognition can be beneficial for several medical diagnosis or treatment plans.

The current main techniques to monitor sleep make use of camera-based systems in addition to various body sensors that can be very uncomfortable during sleep. They allow to collect various data useful to many different diagnosis, but require the patients to spend the night in hospital. Moreover, cameras can suffer from low illumination conditions, occlusions, as well as bad viewpoint angle. Another issue to consider is the violation of the privacy of the users, that forbids to share this kind of data for public use. All these problems can be solved using pressure-based pose detection systems. In particular, commercial pressure sensing mattresses have recently been introduced in the market, allowing for smart home settings. Of course, they can not replace sleep labs, but for sure they represent a much more practical way to monitor sleep. Indeed, different studies have exploited pressure data to perform not only posture recognition [3], [4], but also subject identification [5], that might be very useful for the personalization of smart home experiences; the previously cited studies achieved very promising results, even if all of them were limited to a few main sleeping positions, i.e. supine, left, and right side. What we know to be state of the art is reference [6], whose CNN-based architecture achieves almost 100% accuracy in the identification of subjects and classification of the three main postures.

[†]Department of Physics, University of Padua, email: michelemaria.crudele@studenti.unipd.it

[‡]Department of Physics, University of Padua, email: filippo.ziliotto@studenti.unipd.it

[1]The code of the work can be found at https://github.com/ZiliottoFilippoDev/Human-Data-Analytics

Worse results were reached in the classification of 17 different postures, especially when validating the model with LOSO scheme using augmented data.

As a consequence, in this work we try to achieve higher accuracy in the classification of the 17 postures, building a model more robust to new and more diverse data w.r.t. the one proposed by [6]. We use the same dataset as in [6] and we first feed it to different standard networks, namely CNN, CNN+RNN and CNN+LSTM, where the first CNN blocks are used as automatic feature extractors. We then choose the best one in terms of accuracy as a building block for our own architecture, inspired by *GoogleNet* [7] and making use of inception blocks. All our models are validated using K-Fold and LOSO schemes to obtain more realistic estimates of their performance and are always trained with augmented data to reduce overfitting and increase their ability to generalize to more diverse datasets. Differently, data augmentation is used on the test set only to check the performance of our final model and compare the results with [6]. Posture and subject classification tasks are always learned simultaneously, since [6] showed this to improve the accuracy of both tasks. In this way, we manage to provide a deep learning architecture that would be more suitable to be integrated in software to perform a smart, user-tailored analysis of pressure data, valuable for medical purposes too.

II. Related Work

The recent introduction in the market of pressure sensing mattresses allows for a much more practical monitoring of sleep posture w.r.t. medical tools. Reference [5] used commercial mats to collect the first two publicly-available datasets of pressure sensor data from 13 participants in various sleeping postures. They were then aggregated in three main positions (left, right and supine) to perform a subject identification with a dense network, trained separately for each posture using 18 manually extracted statistical features. Even if not optimal (the average accuracy was slightly above 80%) really promising results were achieved, validating that individuals have a personal sleep pattern.

Those results were improved in [6], where one of the dataset collected in [5] was fed to a CNN-based architecture, in which both subject and posture classification tasks were learned simultaneously. It achieved an accuracy of 100% in the classification of the three main positions (left, right and supine), while obtaining worse results (87% accuracy) with all the 17 different postures, especially when augmented data were used (75.6% accuracy). Regarding subject identification, accuracy was near 100% with the original dataset, while around 90% with augmented data. Those very good results in both posture and subject classification highlight the great value of pressure data, that can provide very valuable information either in smart home and clinical settings.

Given the already perfect accuracy achieved in the classification of the three main positions and of the subjects using original data, in this work we focus on better classifying subjects and all the 17 different postures making use of augmented data. Based on the performance of 3 different standard architectures, we build a more complex model that achieves significantly better results in terms of accuracy either in subject identification and 17 postures recognition when fed with more diverse pressure maps from new subjects.

III. Processing Pipeline

Multitask learning has been successfully used as a method of generalizing a classifier by learning multiple tasks at the same time. Many applications of machine learning exploited it, such as natural language processing [8], speech recognition and computer vision [9], [10].

For our use case, its beneficial effect on the performance has been shown in [6] in a deep sequential Convolutional Neural Network (CNN) architecture. In the first phase of this work, we take that model as inspiration to train three standard deep learning architectures, i.e. CNN, CNN-RNN and CNN-LSTM. The goal is to check if recurrent neural network, on top of a

CNN based model, could improve results even when no temporal input is given.

In the second phase of the study, we exploit the previous analysis to build our final much more complex architecture, inspired by the *inception* learning framework of GoogleNet [7], that we implement from scratch in a simplified version. The main idea behind it is to learn the high, mid and low level features of data in parallel with different inception blocks (Fig. 1). This type of architecture tends to get "wide" rather than "deep", making GPU training a lot more efficient and faster. To prevent the gradient from vanishing, the authors introduced auxiliary classifiers in the middle of the architecture, applying SoftMax to the outputs of the intermediate inception modules. This solution is computationally too expensive to backpropagate with our resources though. For this reason, we decide to apply SoftMax only to the sum of the outputs coming from the three inception blocks we decide to use to classify postures. In parallel, another SoftMax classifies the subjects too.

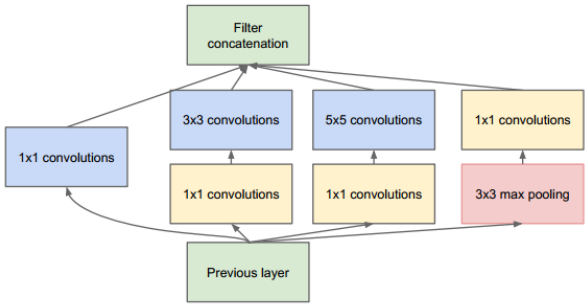

Fig. 1: Inception block architecture implemented throughout our network. Filters of multiple sizes operate at the same level. 1x1 filters are useful to limit the number of input channels before the 3x3 and 5x5 convolutions. The outputs are concatenated and sent to the next inception module.

IV. Signals and Features

The public dataset we work with, PmatData [5], is made up of 18698 pressure maps from 13 participants in 17 different postures. The age of participators was 19-34 years, with a height and weight of 170-186 cm and 63-100 Kg respectively. Pressure data are collected using Vista Medical Force Sensitive Application (FSA) SoftFlex 2048, a commercial mattress that contains 2048 sensors uniformly distributed across a 32x64 grid, with sensors being almost 1 inch apart from each other. Each sensor is able to measure pressures reporting numbers in the range of [0-1000], with a sampling rate of 1Hz. For each participant, 17 different files are provided in tab delimited text format, each one related to a specific posture. Each file contains around 120 frames of recordings ($\sim$ 2 minutes), with 2048 columns representing the 32x64 pressure sensor measurements, and each row being a frame.

In the pre-processing phase we remove the first and the last three frames of each file, since they often contain very noisy images. Then, we remove 14 frames whose total sum of pixel values is less than a specific threshold (found after trial and error method). On the other hand, we also remove 2 frames that contain more than 100 pixels whose value is greater than 1000, that is the maximum number a sensor can register; they are artifacts caused by sensors that are subjected to large pressure values outside of the allowed voltage range. All the previous 16 frames contain very noisy, not significant images, in which no subject posture is recognizable. After that, some outliers are still present in many images though. To deal with this, we apply a 3x3x3 spatio-temporal median filter and then we normalize the pixels dividing their values by 1000, since once again, that is the maximum number a sensor can give. The type of the filter is chosen with a trial and error procedure, selecting the best type and size to reduce the number of artifacts, still maintaining the general aspect of the original frame. An example of the effect of these operations is showed in Fig. 2.

Furthermore, we decide to always train our models with augmented data, in order to improve their ability to generalize to new and perhaps imperfect datasets. Data augmentation is performed almost in the same way as in [6], since eventually we want to compare the results with those obtained in that reference, that is state-of-the-art. We rotate images up to $\pm 30^{\circ}$ with a

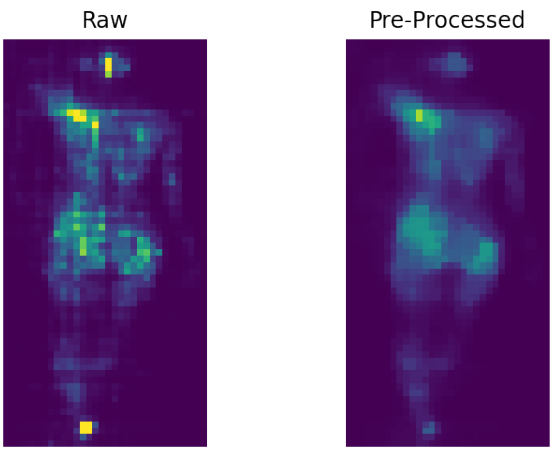

Fig. 2: Illustration of the result of the median filter and normalization pre-processing functions.

probability of 20%. The same probability is used to translate the frames up to $\pm 10\%$ along x and y axis, while we rotate images by $180°$ with a 50% probability. These operations are summarized in Tab. 1.

Regarding feature extraction, it is always done automatically using convolutional blocks.

Finally, given the fact that the dataset is balanced both in subjects and postures, we always split it at random to to obtain training and test sets. The validity of this choice is confirmed by the various iterations of the k-folds, that always output very similar results. The proportion of training and test sets depends on the validation method: in the 10-fold the test set is 10% of the dataset, while when validating with LOSO, it depends on the number of frames associated to the subject used for testing ($\sim 8\%$).

Probability	Process
50%	Rotation by 180°
20%	Rotation by up to ±30°
20%	Horizontal shift by up to ±10%
20%	Vertical shift by up to ±10%

TABLE 1: Data augmentation steps with relative probabilities.

V. Learning Framework

A. Models Architecture

In the first part of our work we compare the performance of different deep models: CNN, CNN+RNN and CNN+LSTM. We do this for choosing the best building blocks to be implemented in our final architecture. In each of those three models, we implement the following CNN-based module, useful to automatically extract significant features from the input (a 32x64 image): 2 Convolutional-BatchNormalization-MaxPooling blocks followed by 2 Convolutional-BatchNormalization blocks. The convolution kernels are 3x3 with a stride of 1 and the number of channels doubles after every convolutional block. LeakyReLu activation function is used before a MaxPooling with 2x2 kernel size. The output of this module is then fed directly to dense layers in the CNN architecture, while in CNN+RNN and CNN+LSTM, three RNN and LSTM blocks are respectively introduced before the dense layers. After a dropout to prevent overfitting, the signal is eventually sent to two Dense layers for parallel classification of posture and subject using SoftMax. These two layers have 13 and 17 units, that are the number of users and sleeping postures in the dataset, respectively. The quantity of channels in the first CNN block, the number of dense layers and their neurons is chosen in such a way to build 3 architectures with more or less the same number of parameters ($\sim 130'000$), to compare them fairly.

With the results obtained from this comparison, we develop our own more complex and accurate architecture, whose learning framework is shown in Fig. 3. It counts about 34 millions parameters and is inspired by the one implemented in [7], with some simplifications. The just cited model is composed of stacked Inception modules. Basically, each module is an image model that aims to approximate an optimal local sparse structure in a CNN. Put simply, it allows to use multiple filter sizes instead of just one in a single image block, which we then concatenate and pass onto the next inception layer [7]. The inception block we use in our implementation is shown in Fig. 1. Strides, padding and kernel filters are left the same as [7]. Going deeper in details, we stack three inception blocks to

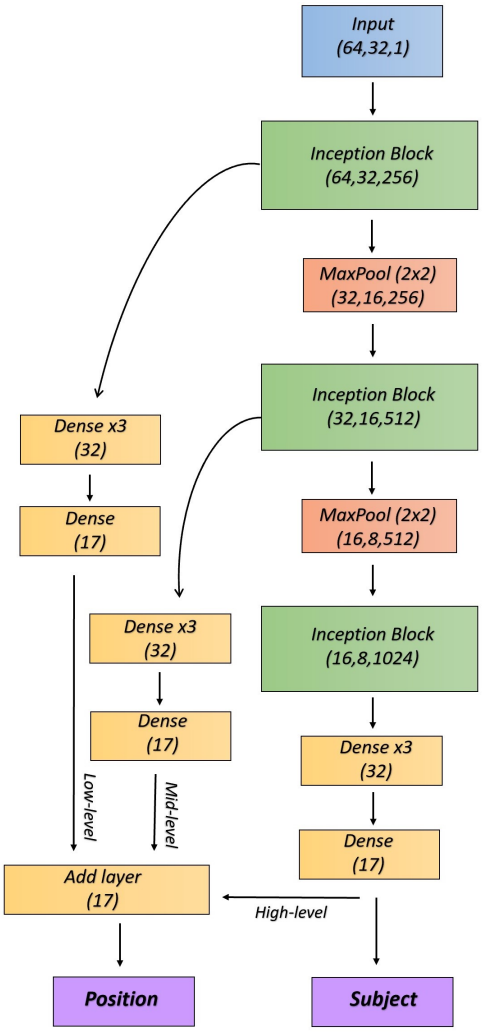

Fig. 3: The architecture of our model. Three convolutional blocks extract low, mid and high level features that are then fed to dense layers. For each level, a 17-neurons dense layer is output. Eventually, they are added together and two SoftMax are used in parallel to classify subjects and posture. To reduce the computational complexity and further encode the input along the way, after every inception block a MaxPool layer is introduced. For regularization, dropout is used in the dense layers.

extract three different feature types all along the network, each one followed by four dense layers blocks. Each block has two times the number of filters of the previous block, starting from the 64 filter in the low level block. The concatenated outputs of these blocks are then added in the end to the related *Add* layer from which we apply (SoftMax). It is useful to notice that actually the network is specifically designed to have outstanding posture recognition, so for subject identification we have only the last levels features that are outputted in parallel.
The input (64,32,1) dimension is reduced throughout the whole network with a MaxPool (2x2) layer after every inception block. The choice of adding together the three feature layers and not backpropagating them singularly is a simple computational trade-off between complexity and training time; still, this models trains in ~ 1 minute per epoch with a *Colab* GPU. Due to lack of hardware resources, a rigorous hyperparameters optimization is neglected.
Finally, all the layers share the same ReLU activation function, we also add a Dropout of 10% before the three final fully connected output layers, to reduce the risk of overfitting.

B. Loss Function

For what concerns the loss function we stick to the one implemented in [6], since we have a multi-output classification. Basically it is a weighted sum of the cross-entropy losses for both subject and posture classifications. The sum is weighted by a hyperparameter λ which is chosen to be 0.5 for k-fold and 0.2 for LOSO validation:

$$\mathcal{L} = \lambda \mathcal{L}_{user} + (1-\lambda)\mathcal{L}_{posture},$$

where

$$\mathcal{L}_{user} = -\sum_{j=1}^{M} \gamma_{ij} logP\left(\gamma_j | I_i\right), \qquad (1)$$

$$\mathcal{L}_{posture} = -\sum_{j=1}^{N} \delta_{ij} logP\left(\delta_j | I_i\right). \qquad (2)$$

We consider a training set composed of tuples $(I, \gamma, \delta,)$, where I is the input pressure map, γ and δ are the user and the corresponding posture respectively. As stated before the last layer of the neural network consists of two *softmax* activations, placed in parallel, with M and N units, where M and N are the number of users and sleeping postures in the dataset.

We use a simple Adam optimizer across 20 training epochs (training for longer would result in useless computational time without significant improvements). A ReduceOnPlateau scheduler is also applied every five increasing loss steps, starting from 10^{-3} as the initial learning rate.

Our Model		
	Block	Parameters
1^{st} Module	Inception	140'288
1^{st} Module	Dense	16'785'600
	MaxPool (2x2)	
2^{nd} Module	Inception	951'040
2^{nd} Module	Dense	8'389'504
	MaxPool (2x2)	
3^{rd} Module	Inception	3'802'624
3^{rd} Module	Dense	4'202'688

TABLE 2: Details of the parameters of our architecture. The three modules learn three different levels of features. The total number of parameters is $34'262'752$. Each Dense block is made up of three fully connected 64x1 layers and a final 17x1 (one neuron for each posture) layer which is eventually added to the outputs of the other blocks.

VI. Results

The results obtained in the state-of-the-art work [6] with the same dataset we use in this study are already optimal in the classification of the subjects and the three main in-bed postures, i.e. supine, left and right side. As a consequence, in this study we focus on the classification of all the 17 postures collected in the dataset, in addition to the identification of the subjects, since [6] showed that learning both tasks simultaneously is beneficial in terms of accuracy.

The first result of our study is about the performance of three $130'000$ parameters architectures, namely CNN, CNN+RNN and CNN+LSTM. In Tab. 3 and Tab. 4 we show their percentage accuracy in the classification of the 17 in-bed postures and the 13 subjects, respectively. With LOSO, subject classification in not possible, since data from one single subject are used in the test set. From those tables, we can see how including RNNs or LSTMs after the CNN blocks does not improve the accuracy in both tasks. This is sensible, since RNNs (and LSTMs) are thought for sequential data, while we are dealing with static images. As a consequence, we do not use them in our final architecture.

17 IN-BED POSTURES ACCURACY (in %)

Architecture	10-fold	LOSO
CNN	99.67 ± 0.37	80.1 ± 8.7
CNN + RNN	98.79 ± 2.19	74.3 ± 6.9
CNN + LSTM	94.97 ± 8.52	74.5 ± 9.5

TABLE 3: Accuracy of different architectures (all with $\sim 130'000$ parameters) for 17 postures classification. The 10-fold is repeated 2 times. Augmented data is used only for training.

13 SUBJECTS ACCURACY (in %)

Architecture	10-fold
CNN	99.33 ± 2.11
CNN + RNN	98.6 ± 2.0
CNN + LSTM	92.56 ± 9.73

TABLE 4: Accuracy of different architectures (all with $\sim 130'000$ parameters) for 13 subjects classification. The 10-fold is repeated 2 times. Augmented data is used only for training.

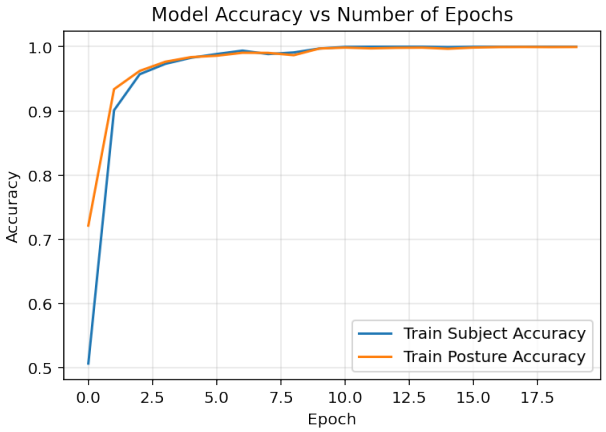

Fig. 4: Task specific accuracy as a function of the number of epochs for our proposed model.

When tested on non-augmented data, our model achieves a 100% accuracy both in 13 subjects and 17 postures classification with a 10-fold validation scheme. In Fig. 4 we presents

an example of the learning curves obtained during of that procedure. It can be seen that our model quickly learns both tasks and converges to a steady state. With LOSO cross-validation instead, it approaches 88% accuracy in posture classification (Fig. 5). This value is only slightly higher than the one achieved in [6] with the same validation scheme (Tab. 5). Still, we stress once more the fact that we are using augmented data for training, while 87% accuracy was obtained in [6] by training with original data. In this way, we build a model that is more robust to new and imperfect data w.r.t. [6]. Indeed, using augmented data also for testing, our accuracy decreases by only $\sim 2\%$ both with k-fold and LOSO, while it drops by $\sim 10\%$ in [6], as shown in Tab. 5 and Tab. 6.

Overall, from those tables we can see that we manage to build a model that performs $\sim 10\%$ better than the state-of-the-art in both tasks when using augmented data for testing. In particular, the improvement in accuracy obtained with LOSO suggests that our model would be more robust to diverse data coming from new subjects, making it more suitable to be integrated in clinical or smart home settings.

17 IN-BED POSTURES ACCURACY (in %)

Architecture	10-fold *(a.t.)*	LOSO	LOSO *(a.t.)*
Our Model	98.6	87.7	85.4
2019 Model [6]	93.2	87.0	75.6

TABLE 5: Our model compared to the one developed in [6]. "*(a.t.)*" in column names means that augmented data is used also for testing.

13 SUBJECTS ACCURACY (in %)

Architecture	10-fold	10-fold *(a.t.)*
Our Model	100	98.0
2019 Model [6]	100	89.7

TABLE 6: Our model compared to the one developed in [6]. The last column refers to the case in which augmented data is used also for testing.

The dataset we work with in this study is almost perfectly balanced both in subjects and postures, so the accuracy is a good metric to analyze the performance. However, we tried to

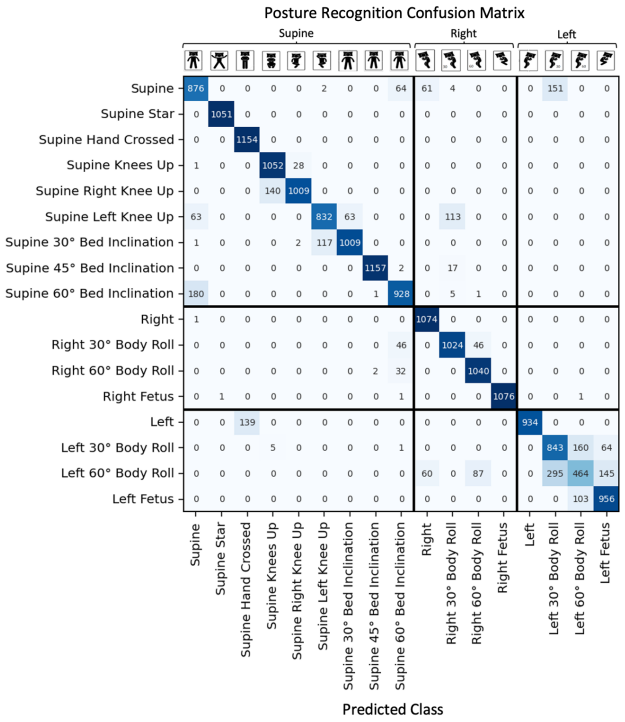

Fig. 5: Confusion matrix of our model with the LOSO validation scheme and data augmentation only on the training set. Predicted labels are at the bottom, true labels on the left. We see how the majority of the misclassified sub-postures fall within the correct main position (supine, right, left). The others are probably due to very strange postures from specific subjects, since they are in the order of 120, that is the average number of frames collected for each posture.

check also other metrics, i.e. precision, recall, and F1-score, verifying that they are always very similar to the accuracy and near 100% when splitting the dataset in training and test set at random.

VII. Concluding Remarks

In this work, a deep CNN inception based framework with parallel blocks exploiting different feature levels is proposed for classification of patients and 17 in-bed postures making use of data collected with a pressure sensing mattress. We build an efficient and highly accurate model that learns both tasks simultaneously and outperforms by $\sim 10\%$ the state-of-the-art model in in posture recognition tested on augmented data. Our model also achieves a $\sim 10\%$ higher

accuracy in the subject identification with KFold validation tested on augmented data, thus proving a robust generalization to previously unseen data and valuable information, which can be deployed in smart home and clinical settings. Future works could focus on hyperparameters optimization, that we did not perform rigorously due to hardware limitations.

REFERENCES

[1] C.H.Lee, D.K.Kim, S.Y.Kim, C.-S.Rhee, and T.-B.Won, "Changes in site of obstruction in obstructive sleep apnea patients according to sleep position: a dise study," *The Laryngoscope*, vol. 125, no. 1, pp. 248–254, 2015.

[2] J. Black, M. M. Baharestani, J. Cuddigan, B. Dorner, L. Edsberg, D. Langemo, M. E. Posthauer, C. Ratliff, G. Taler, and N. P. U. A. Panel, "National pressure ulcer advisory panel's updated pressure ulcer staging system," *Advances in Skin & Wound Care*, vol. 20, no. 5, pp. 269–274, 2007.

[3] S. Boughorbel, F. Bruekers, and K. de Groot, "Pressure-sensor system for sleeping-posture classification," *Measuring Behavior*, p. 358, 2012.

[4] Q. Sun, E. Gonzalez, and Y. Sun, "On bed posture recognition with pressure sensor array system," *IEEE Sensors*, pp. 1–3, 2016.

[5] M. B. Pouyan, J. Birjandtalab, M. Heydarzadeh, M. Nourani, and S. Ostadabbas, "A pressure map dataset for posture and subject analytics," in *IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, (Orlando, FL, US), 2017.

[6] V. Davoodnia and A. Etemad, "Identity and Posture Recognition in Smart Beds with Deep Multitask Learning," in *IEEE International Conference on Systems, Man and Cybernetics (SMC)*, (Bari, Italy), 2019.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[8] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, (New York, NY, USA), pp. 160–167, Association for Computing Machinery, 2008.

[9] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), (Cham), pp. 94–108, Springer International Publishing, 2014.

[10] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.