Estimation of the pressure map of a human body on a mattress from infrared images

Gabriel Gausachs Fernández de los Ronderos

Abstract—Despite the successes of deep neural networks in visual tasks, their application in practical medical settings remains limited. One of the persistent challenges is measuring the pressure of a human body on a mattress, crucial information for preventing pressure ulcers in patients hospitalized for extended periods. Traditional technologies, such as pressure sensors, have limitations such as cost and the need for calibration. In this context, we propose using infrared images to estimate the pressure map. This approach, involving the implementation of a U-NET network and the use of the SLP (*Simultaneously-collected multimodal Lying Pose*) dataset for training, promises an effective and economical alternative for pressure monitoring in hospitalized patients.

Index Terms—Neural Network, IR image, Pressure Map, ulcers

1 Introduction – Context of the Work

Currently, with technological advances and the continuous emergence of artificial intelligence and computer vision, we are seeing more and more practical medical applications in these fields. However, their use is still not very common, and there are many issues and problems to address.

This project focuses on estimating the pressure map of a human body on a mattress from infrared images. In many cases, patients who spend a lot of time bedridden in hospitals cannot reposition themselves, and doctors cannot constantly monitor all of them. When a patient remains in the same position on a bed for a prolonged period, there are areas of their body where the bone presses the skin against an external surface, in this case, a mattress. This can cause the appearance of skin lesions called ulcers.

The appearance of ulcers can be avoided if the patient changes position regularly and does not maintain constant pressure on different areas of their body.

Today, to prevent ulcers, there are robotic beds that, among other functions, can tilt or move automatically when they detect constant pressure in certain areas of the body. Therefore, these beds must be capable of automatically estimating the body's pressure on the mattress. Various sensors can be used, such as RGB cameras, depth cameras, pressure sensors, and sensor fusion methods. However, pressure sensors are very expensive, prone to errors, and require calibration.

This problem has long been a challenging task. Over the years, studies have found solutions to this problem. Shuangjun Liu and Sarah Ostadabbas [1] first collected the SLP (*Simultaneously-collected multimodal Lying Pose)* dataset and worked on predicting the posture of a person using infrared and RGB images. Subsequently, they studied the prediction of pressure maps [2]. Additionally, they explored the possibility of incorporating into the model a vector of phyisical data to achieve better performance. On the other hand, Henry M. Clever [3] predicts pressure maps from depth images and augmented data.

Therefore, this work attempts to provide a similar approach and achieve the objective with more affordable methods than using sensors. Specifically, infrared images are used as input to a UNET network that generates the body's pressure map.

On the other hand, an infrared image alone may not be sufficient to estimate a physical quantity, such as pressure. Two people lying in the same position might have very similar infrared images. However, if one of them weighs twice as much as the other, the pressure they exert will not be the same in both cases. Therefore, in addition to the infrared image, this project also explores including an information vector with data such as weight, height, etc., as input to the network.

In this project, the SLP dataset is used to train and test the created model. The Dataset section explains in more detail what data the dataset contains and how it is organized.

As previously mentioned, the main objective of this work is to create a method using computer vision and deep learning to estimate the pressure map of a human body on a mattress from infrared images. Within this project, however, several more specific objectives are identified:

- Understand, process, and clean the SLP dataset.
- Appropriately select the best loss functions for the proposed problem.

- *Contact email: 1604373@uab.cat*
- *Project guided by: Vanessa Moreno Font (Area of Computer Architecture and Technology*
- *With collaboration of: David Castells Rufas (Area of Computer Architecture and Technology)*
- *Year 2023/24*

- Develop and train a UNET neural network.
- Evaluate the model's performance using suitable metrics.
- Optimize the model parameters to improve its accuracy and performance.
- Investigate the robustness and generalization of the trained model against variations in environmental conditions, body postures, or other factors that may influence the results.

2 DATASET

2.1 Simultaneously-collected multimodal Lying Pose (SLP)

The dataset for this project was collected by the Augmented Cognition Lab (ACLAB) at Northeastern University in Boston, Massachusetts [1]. It contains images of different patients lying in beds in various postures. The images are in 4 modalities, specified later, and there is also a file with relevant physical information for each patient. This dataset consists of 2 data collections: Hospital setting, which includes data from 7 patients, and Home setting, with 102 patients aged between 20 and 40 years. In this project is used the Home setting collection.

Each patient is positioned according to 3 categories: left side, right side, and supine (lying flat on their back). For each category, 15 postures were recorded. Additionally, each patient was recorded under 3 layering conditions: uncover, where the patient is not covered by any blanket; cover1, where the patient is covered by a 1 mm thick blanket; and cover2, with a 3 mm thick blanket. In total, there are 13,770 samples, each in 4 different modalities: RGB, using a regular camera; LWIR (Longwave Infrared) using a FLIR IR camera; DEPTH, using Kinect v2; and the pressure map, using a Tekscan Pressure Sensing Map.

The project focuses only on the LWIR images and the Pressure Map, as the goal is to predict the pressure map from an infrared image. Specifically, these images are structured in arrays.

2.2 Resizing and Standardization of LWIR

Before training a model, it's important to preprocess the input data to achieve optimal and effective performance. In this project, it's first identified that the arrays have different dimensions. On one hand, the LWIR arrays are of size 160x120, whereas the Pressure Map arrays are of size 192x84. This can be a problem because it's important that the pixels of the input arrays are aligned with those of the output. Therefore, the first data preprocessing step is to resize the LWIR arrays.

Observing the LWIR arrays, it's found that they cover a larger area than the bed where the patient is lying, whereas in the pressure maps, the image boundaries are practically the mattress boundaries. That's why a cropping operation is applied to the LWIR arrays where the center of the bed is defined as the center, and then resized to 192x84. This way, both arrays have the same size, and their pixels are aligned.

Finally, to achieve stability in the training of the model and avoid convergence problems, it's important to have the input data within a specific range and centered around a certain mean. Consequently, the LWIR arrays are scaled to have values in the range [0,1] and then undergo a standardization process.

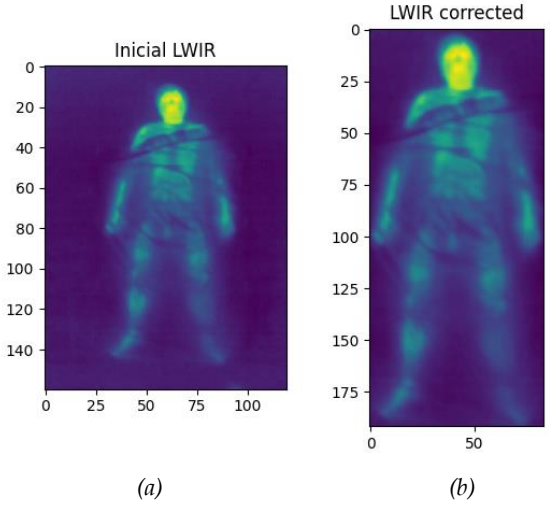


Fig 1: (a) Initial LWIR image. (b) Transformed LWIR image

2.3 Checking and processing the ground truth

The goal of this project is to predict the total pressure a patient exerts on a mattress. For this reason, the sum of the values of each pixel in the Pressure Map should represent the total pressure of the patient, and by applying the formula that relates mass with pressure and surface area, the corresponding mass of the patient should be obtained. Both the weight of the patient and the surface per pixel are available.

Therefore, the next step is to verify the ground truth of the data. It is concluded that the patient's weight obtained from the pressure values is much higher than the actual weight. For example, for patient 1, the weight gives values ranging from 450 kg to 700 kg when the real weight is 80 kg. Given this, a problem that arises is the correction of the ground truth data.

To address this issue, the dataset provides a calibration file for each patient. This file is an array of size (3, 45) where each row corresponds to a condition (uncover, cover1, and cover2) and each column corresponds to an image number. Therefore, the solution proposed by the dataset is to multiply the pixel values by the corresponding calibration value. This solution resolves the discrepancy found between the weight obtained from the Pressure Map and the actual weight, with an approximate difference of 250 grams. However, what actually happens is that each Pressure Map is multiplied by a specific value to yield the

correct weight, resulting in an unrealistic Pressure Map because each image has been independently adjusted to match the weight.

That's why the initial proposal in this project is to add a fully connected layer to the output of the network so that each pixel value of the model's output is multiplied by a learned value, which will be the "correction" to ensure the total weight matches the actual weight.

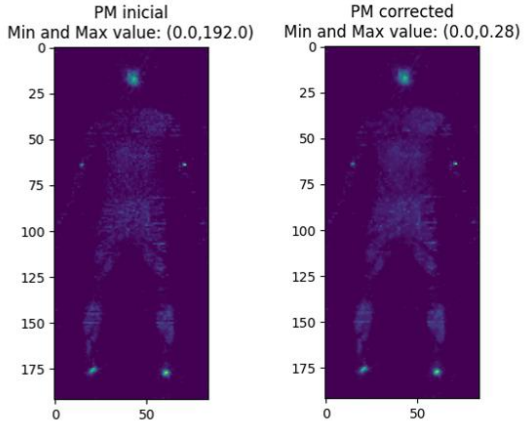
It seems that the sensor used to measure pressure has not worked consistently or accurately. Anomalies have been observed in the pressure map images, where there are areas with minimum values surrounded by elevated values. This phenomenon lacks coherence, as it does not always occur in the same positions of the images. This indicates that the sensor may have failed to accurately detect pressure in some areas.

Due to this conclusion, the final proposal of this project to achieve correct ground truth is to apply a median filter to the pressure map (PM) [4], but only changing those pixel values where the value with the median filter is higher than the already given value (1).

(1)

$$I(i,j) = \max\left(I(i,j), Median\ Filter\ \left(I(i,j)\right)\right)$$

And once the new array is obtained, normalize it so that the total sum of the values corresponds to the pressure equivalent to the weight of the patient. Then, the values are scaled to have values also in the range [0,1]. In figure 2, it can be seen the transformation of the PM. It can be observed that in the PM corrected the values are between 0 and 1. In this example, the màximum value is not 1 because the pressure is not all concentrated in a singular píxel, it is more distributed.



(a) (b)

Fig 2: (a) Initial PM with its max and min values (b) PM corrected after transformations with its max and min values

2.3 Data imbalance

Data imbalance occurs when there's a significant disproportion in sample numbers across different classes, which can hinder machine learning models from effectively learning minority classes. In a project aiming to predict high-pressure areas on a mattress to prevent pressure ulcers, imbalanced data—where low-pressure areas dominate—poses challenges, as high-pressure areas become the minority class.

Below (Figure 3), two histograms are observed, one for LWIR and the other for PM, from an example of a patient. It can be observed that in the case of PM, the data does not follow a normal distribution, and most of the values are concentrated in 0.00.

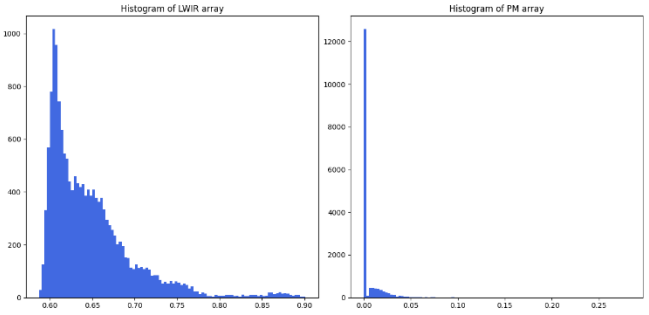


Fig 3: (a) LWIR imatge histogram. (b) PM histogram

Imbalanced data can skew model predictions towards the majority class, especially if crucial information resides in the minority class. For instance, if the model predicts all values as zeros, it may achieve low loss by matching the majority of data labels, but it fails to learn from minority classes. Addressing this, a new loss function can be defined to mitigate data imbalance by giving more weight to minority class samples during training, thereby promoting accurate identification of critical areas.

2.5 Physical data

Regarding the physical data of each patient, there is a CSV file with the following variables:

List of the phyisical variables

Variable	Unit	Range
Weight	kg	[148, 184]
Height	cm	[47, 83]
Bust	cm	[65, 108]
Waist	cm	[62, 112]
Hip	cm	[82, 118]
R lower arm	cm	[23, 37]
R upper arm	cm	[18, 30]
R upper leg	cm	[38, 64]
R lower leg	cm	[28, 43]

Table 1 : List of the physical variables

This vector is scaled to have values between 0 and 1 using

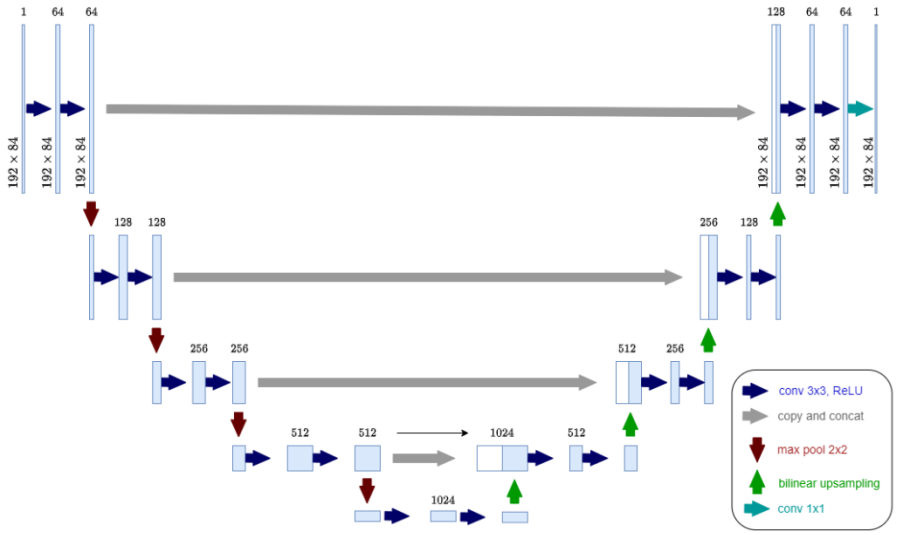


Fig 4: U-Net arquitecture [7]

the MinMaxScaler tool from the sklearn library, a library in Python that provides many unsupervised and supervised learning algorithms. It's built upon some of the technology you might already be familiar with, like NumPy, pandas, and Matplotlib.

3 MODEL

The architecture of the model is the U-Net [5]. The U-Net architecture is a very popular convolutional neural network (CNN) used in many biomedical tasks. The U-Net architecture gets its name from its shape, which resembles the letter "U". It is composed of two main parts: the "encoder" (top part of the "U") and the "decoder" (bottom part of the "U"). In the encoder, higher-level features of the input are captured, and the spatial dimension of the input is reduced. In the decoder, the lost spatial information during compression is recovered, and an enlarged version of the input is reconstructed. The main feature of the U-Net is that in the different layers of the decoder, the previous layer and also the equivalent output layer in the encoder are received. This way, information at different scales is leveraged, improving the model's accuracy [5]-[6].

As seen in Figure 4, the encoder of this project's U-Net consists of 4 blocks of 2 convolutional layers each and maxpooling, where each layer of each block is accompanied by Batch normalization and ReLU activations. At the end of the encoder, there is a final block of 2 convolutional layers where the data is transformed to have 1024 channels. As for the decoder, the path is reversed but adding information from the corresponding descending layer. In total, the final model has approximately 31M parameters.

This network uses the He weight initialization [5] by default, where the weights are assigned from a normal distribution with a mean of 0 and a specific standard deviation depending on the number of input units of each layer.

As mentioned earlier, in this project, it is explored incorporating a vector of physical data from the patients into the network to achieve better performance. One possible approach, devised and used by Shuangjun Liu and Sarah Ostadabbas [2], is to separately encode the arrays and the information vectors, concatenate them at the bottleneck, and then jointly decode them.

3 LOSSES FUNCTIONS

In this project, one new loss function, called High Values Loss (HVLoss), is experimented with, in addition to the Mean Squared Error (MSE) and the SSIM (Structural Similarity Index Mesure).

The SSIM is a perceptual metric used to measure the similarity between two images. Unlike traditional metrics like Mean Squared Error (MSE), which focus on pixel-wise differences, SSIM considers changes in structural information, luminance, and contrast. This metric is very common when dealing with images [8].

3.1 High Values Loss (HVLoss)

The proposed loss function is the HVLoss. This function gives higher weight to pixels with high values. The higher the pixel value, the more important the error in that pixel will be. With this function, the aim is to address the data imbalance (2).

(2)

$$HVLoss = \frac{1}{n}\sum_{i=0}^{M}\sum_{j=0}^{N}\left(\hat{y}(i,j) - y(i,j)\right)^2 * h(y(i,j))$$

Where

$$h(y(i,j) = \frac{y(i,j) - min(y)}{max(y) - min(y)} * 0{,}9 + 0{,}1$$

Where the min and max values are from each batch.

With this equation, each squared difference is multiplied by a number between 0.1 and 1, where the higher the pixel value, the higher this number will be.

4 METRICS

A common way to evaluate a regression model is by calculating the Mean Squared Error (MSE), where each pixel contributes equally to the calculation. However, in this project, it is proposed to evaluate the model's performance with three additional metrics: PerCS (Percentage Correct Sensing), MSEeff (Mean Squared Error with effective matrix) and the popular SSIM (Structural Similarity Index Measure). In this section it is only explained the PerCS and the MSEeff metrics.

4.1 Percentage Correct Sensing (PerCS)

In this regression problem, the difference between predicted values within a predefined range can be considered as a correct result. Additionally, the few outliers that may exist can contribute to a high MSE, even if the majority of estimated values are correct. That's why this new metric is proposed, where the total number of pixels where the difference is less than 2.5% of the maximum value in the batch is divided by the total number of values in the batch (4).

(4)

$$PerCs = \frac{|E < \varepsilon|}{|E|}$$

In this context, $|.|$ represents the cardinality, E refers to the difference between the model's output and the target, and ε is the defined threshold.

4.2 Mean Squared Error with effective matrix (MSEeff)

This metric is a version of MSE but focuses solely on evaluating the model on values where there is pressure and, therefore, where the patient's body is located. (5)

(5)

$$MSE_{eff} = \frac{1}{n}\sum_{i=1}^{M}\sum_{j=0}^{N}\left(\hat{y}(i,j) - y(i,j)\right)^2 \quad when\ y(i,j) > 0.05$$

In the equation, only the pixels where the value is greater than 0.05 are filtered in order to evaluate only those where there is pressure.

5 EXPERIMENTAL DEVELOPMENT

Once the different loss functions and metrics to be used for evaluating are defined, the models are trained and their performances analyzed. The dataset is divided into three subsets: training, validation, and test, ensuring that each subset includes data from each patient and each condition (uncover, cover1, cover2). The training set consists of 10,098 data points, the validation set has 2,754, and the test set includes the remaining 918 data points. Both in training and validation, a batch size of 32 is used.

For training, the Adam algorithm [9] is used as the optimizer, with an initial learning rate of 0.0002. The models are trained for a maximum of 100 epochs, but with an early stopping strategy with a patience of 10 epochs to prevent overfitting. The tool Wandb.ai is used to record the losses and metric results in each training session [10]-[11].

5.1 Training with the 3 different losses

The first 3 training sessions utilize the loss functions MSE (Mean Squared Error), HVLoss and SSIMLoss, respectively.

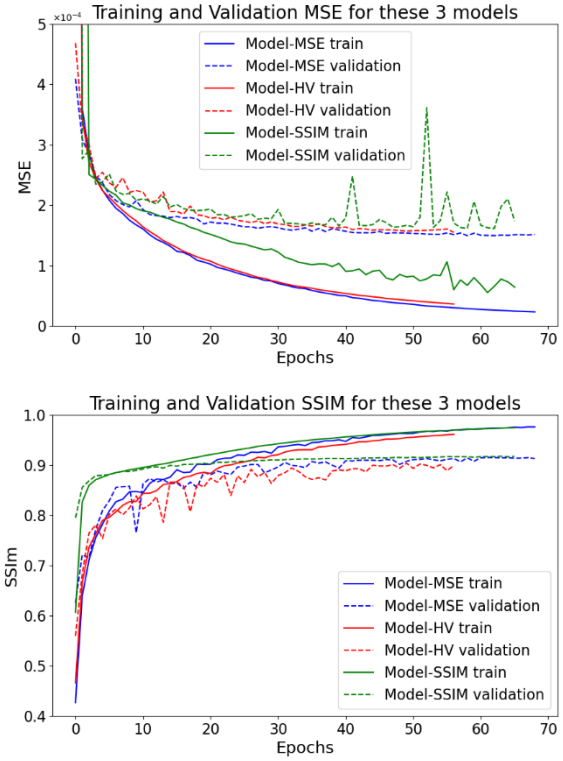


Fig 5: Training and Validation for SSIM and MSE in 3 models, each trained with different loss

Analyzing the results shown in Figure 5, it can be observed that the performance in MSE using MSE as a loss (blue) is better than using High Value Loss (red) and SSIM (green), both in the training and validation sets. On the other hand, in the graph below, related to SSIM metric, it can be observed that the model using HV is the worst but both the MSE and SSIM model have a similar performance. Additionally, the concept of overfitting is evident in both training sessions. Regarding the MSE model, there is a difference between the curves of the training set and the validation set, both in the MSE metric and in the SSIM. For the training set, an MSE of 0.231×10^{-4} and a SSIM of 97.70% are achieved. On the other hand, in the validation set, an

MSE of 1.49×10^{-4} and a SSIM of 91.42% are obtained.

After studying the results of these 3 training sessions, it is concluded that overfitting appears in the training. Overfitting occurs when a model fits too closely to the training data to the point of capturing the noise or specific peculiarities of these data. This can result in a loss of generalization ability, as the model cannot properly generalize to new unseen data, maintaining or increasing the error in the test set.

One reason for the occurrence of overfitting is that the model has very high complexity. When a model has an excessive number of parameters or layers, it may lead to the model memorizing the training examples rather than learning generalizable patterns. The three previous models are trained with a UNet network with over 31M parameters, a considerably high number. In the bottleneck layer of the model, which is the narrowest part of the network, 512 input features are used, which are transformed into 1024 output channels. This high dimensionality at the bottleneck may be the cause of overfitting.

5.2 Training with diferent number of parameters

Therefore, the aim is to train the model with different numbers of parameters to achieve a reduction in overfitting and better generalization of the model. In these training sessions, MSE Loss is used with the same configurations as in the previous training sessions.

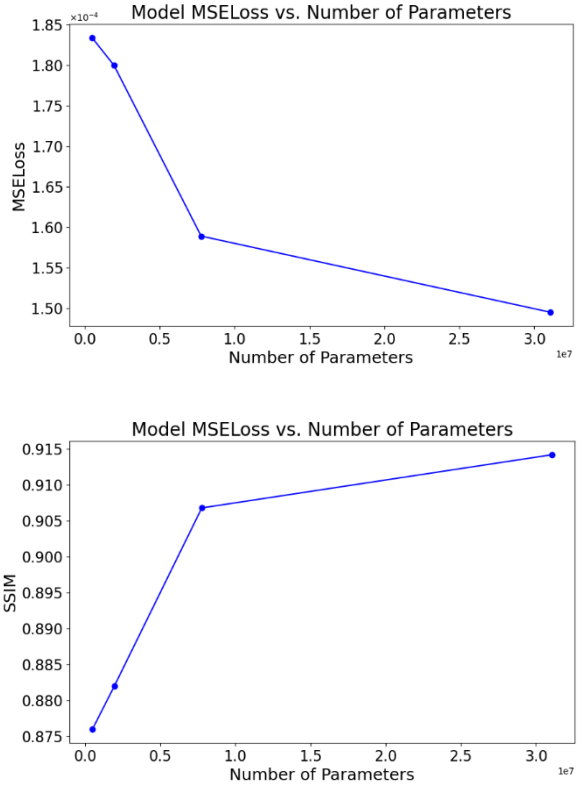


Fig 6: MSE and SSIM for each model trained with different number of parameters

In these graphs, the MSE Loss and the SSIM of the validation set in different training sessions is observed, each with a different number of parameters.

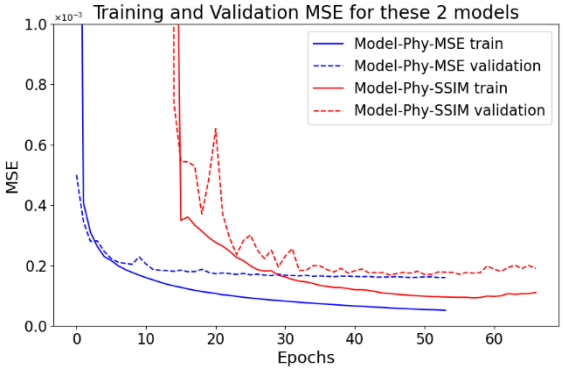
At first glance, it can be seen that the higher the number of parameters, the lower the MSE and the higher the SSIM. Upon closer inspection, there is a considerable increase in MSE when reducing the number of parameters from 7M to 2M and also a decrease related to the SSIM. On the other hand, the difference between the training with 7M and 31M parameters is not very significant. With this analysis, the goal was to find a balance between reducing overfitting without significantly increasing the MSE Loss and decresing the SSIM. Therefore, by calculating the difference between the training set and the validation set in the models with 7M and 31M parameters, a reduced overfitting of 12.43% was obtained with only a 9.4×10^{-6} increase, that represents a 6.28%.

This result demonstrates an effective trade-off: by reducing the number of parameters to 7M, the model maintains a similar validation performance to the more complex 31M parameter model, while achieving significantly better generalization (less overfitting). Therefore, it is studied the possibility to increase a bit the loss while reducing the overfitting by reducing the number of parameters to 7M.

5.3 Incorporation of the physical data vector

In this project, the incorporation of a vector containing physical data of the patients into the network is also studied. This vector is encoded with linear layers and then concatenated with the arrays at the bottleneck of the network for joint decoding. [11]-[12]. Various trainings with different configurations are carried out. First, the model is trained using the MSE loss function as previously done and then with the SSIM too.

In Figure 7, the curves of the two models trained with the incorporated physical vector are shown. It can be observed that both trainings are similar, but obviously it's clear that the model using SSIM as loss has a better performance in the SSIM metric but a worse one in the MSE. In both models, an approximate MSE of 0.17×10^{-3} and a SSIM of 91.50% are achieved. These results are not far better than the results obtained with the model without the physical data. The main reason lies in the preprocessing of



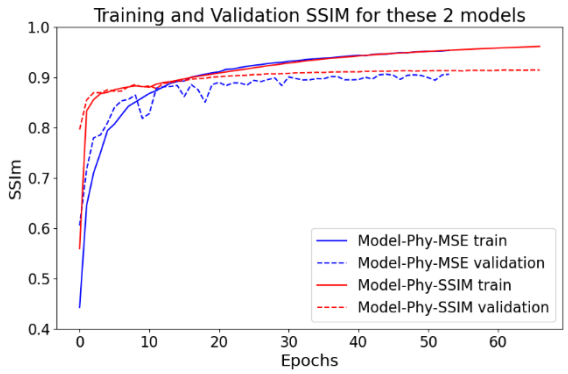


Fig 7: MSE and SSIM for the two models trained with physical vector added

the pressure maps. As previously explained, the pressure maps are normalized so that the total sum of the values equals the pressure corresponding to the patient's weight.

Therefore, this process already takes into account the patient's main physical characteristic, their weight. It is concluded that the proposal to incorporate a physical vector into the model is not optimal.

5.4 Final results

Finally, after training different models and analyzing them, this table summarizes the results obtained with each configuration explored in each metric used:

Final Results

Model	MSE_{e-3}	$MSEeff_{e-3}$	$PerCS_{2,5}$	SSIM
U-Net MSE	**0.149**	6.14	0.9063	0.9142
U-Net HV	0.159	**5.85**	0.8937	0.8981
U-Net SSIM	0.221	6.65	**0.9071**	**0.9170**
U-Net 7M params	0.158	6.43	0.9005	0.9072
U-Net Phy MSE	0.155	6.36	0.9022	0.9112
U-Net Phy SSIM	0.176	6.62	0.9068	0.9162

Table 2 : Final results of each model in each metric

First of all, looking at Table 2, it is concluded that all the trainings have achieved very similar results.

Regarding the Mean Squared Error (MSE), the U-Net MSE model, trained with 31M paràmetres, obtained the best performance, but the second is also using MSE but with a more generalization performance.

If it is considered the MSE only in the pressure zones, MSEeff, the best model was the U-Net HV. This is understandable since this loss focuses specifically on the areas where the pressure map values are higher, that is, in the pressure zones. However, the overall MSE is higher, so this model is more precise in the pressure zones but has more difficulties in correctly detecting where these zones are.

What's more, it is observed that the U-Net SSIM model has the best performance in SSIM metric and PerCS but, as it is seen in the the graphs before (figure 5), the MSE is way worse than the U-Net MSE.

Finally, the results of the models trained with the physical vector are observed, and it is concluded that the values obtained are very similar. Therefore, the possibility of incorporating a vector into the network is discarded since a better result is not achieved. Additionally, not incorporating physical data can be beneficial for using the model in another environment or with another dataset where physical data is not available.

Therefore, after the different trainings and the different approaches tried, it is concluded that for this project the best model is the U-Net MSE and without incorporating the physical data vector.

6 Tests

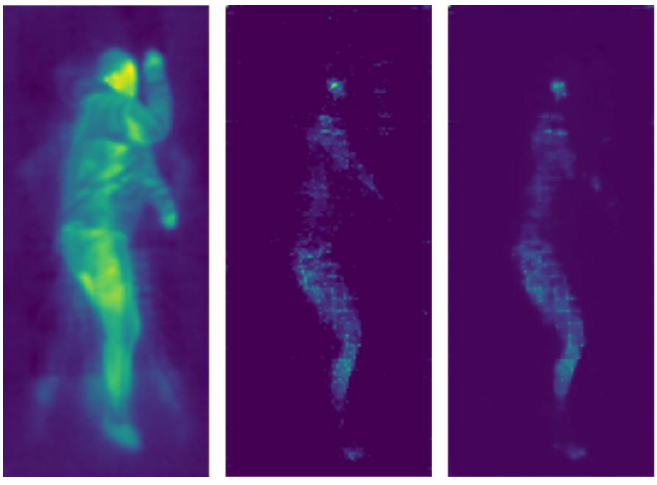
Once the final model is defined, the testing phase of the model with the test dataset is carried out. The performance of the model for different conditions is studied to analyze the generalization and variability of the proposed model.

Regarding the different conditions (uncover, cover1, cover2), a table (table 3) with the values of the metrics used for each condition is provided. Below, in Figure 8, the input image, the target, and the model output for each condition can be observed.

Test Results

Conditions	MSE_{e-3}	$MSEeff_{e-3}$	$PerCS_{2,5}$	SSIM
Uncover	0.145	5.96	0.9086	0.9151
Cover 1	0.150	6.11	0.9039	0.9134
Cover 2	0.144	5.98	0.9058	0.9145

Table 3 : Test results of each condition in each metric



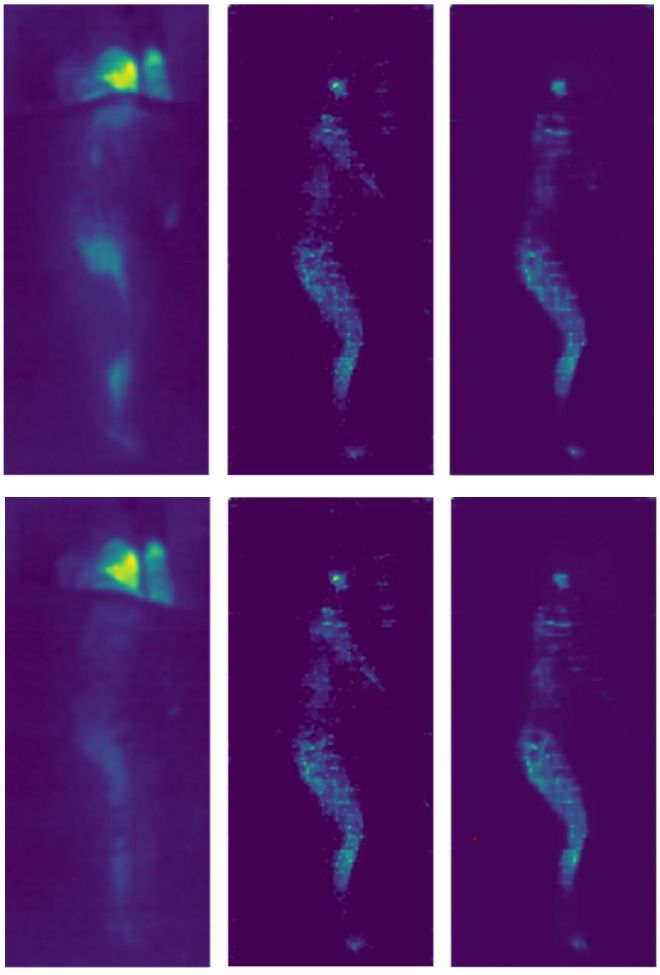


Fig 8: Example of Input, Target and Output of the model en each condition (uncover, cover1, cover2)

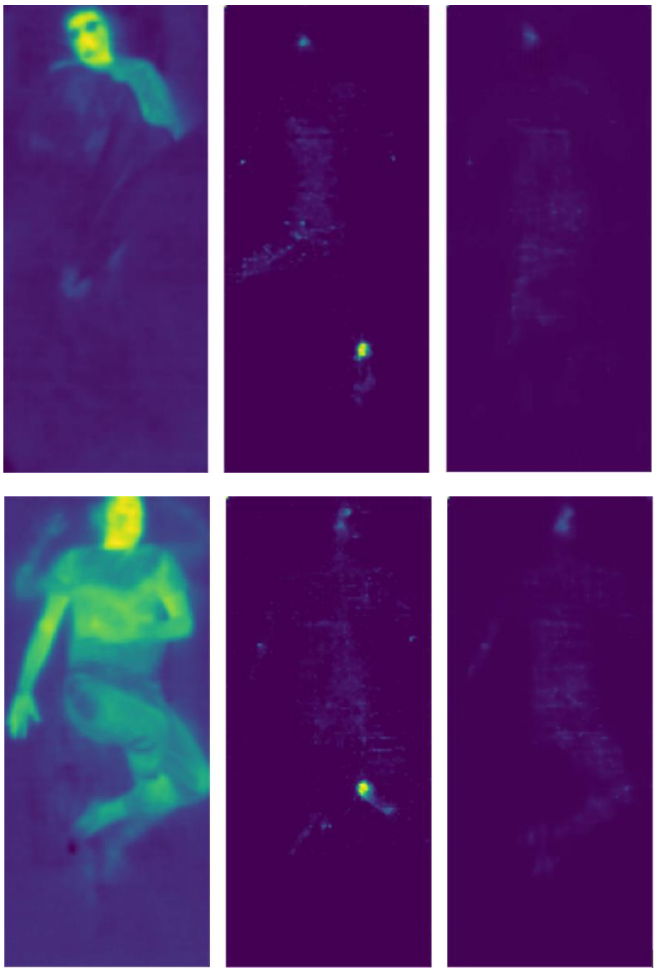


Fig 9: Input, target and output of the 2 worst prediccions

At first glance, it can be observed that the model predicts the pressure maps well. Upon closer inspection, it can be noted that the model outputs are more blurred than the targets, indicating that the model's precision is not perfect. Furthermore, in the table of metric results, it can be concluded that the model behaves very similarly in each condition. This is because, in partitioning the data into training, validation, and test sets, data from each condition was included in each subset to allow the model to learn different patterns.

Also, in this project the worst predictions made by the defined model are studied. Figure 9 shows the two worst predictions of the model with their inputs, targets and outputs.

In figure 9 it can be observed that both cases are very similar. In both cases, in the patient's pressure map there is a point with a very high pressure and much higher than the rest of the points. In these cases, it can be seen that the model is not accurate and is not able to predict this point of maximum pressure. This may be due to the fact that the pressure of most of the pressure maps that the model has trained is distributed among the different parts of the patient's body. In order to get the model to predict these maximum pressure points, the model would have to be trained with more similar cases in order to understand its characteristics better. In both cases, the MSE is about 0.6×10^{-3}.

7 Experimenting with different data

After testing the model and analyzing the results, the model is evaluated with additional data collected in a different environment using different sensors, thanks to the Autonomous University of Barcelona. The objective is to assess the model's generalization and robustness, specifically its ability to accurately predict Pressure Maps in varied settings. This dataset comprises 21 patients with a total of 30 images, including both infrared (IR) images and Pressure Maps (PM).

7.1 Data Preprocessing

To prepare the data for optimal model input, several transformations are applied. For IR images, they are rotated, cropped to center around the bed's center, and resized to dimensions of (192, 84), consistent with the preprocessing applied to the SLP dataset. Notably, the IR image values are already within the range of [0, 1], thus requiring no additional scaling. In contrast, Pressure Maps, provided as CSV files, are rotated, scaled to ensure values ranged between 0 and 1, and resized to the specified

dimensions of (192, 84). Figure 10 illustrates both the original and transformed IR images alongside the PM.

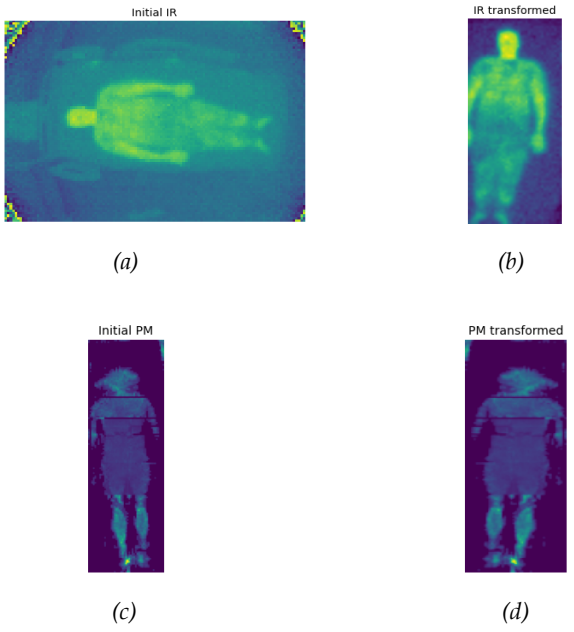


Fig 10: (a) Initial IR, (b) IR transformed, (c) Initial PM, (d) PM transformed

In Figure 10, two observations stand out. Firstly, the IR image appears notably noisy and lacks clarity. Secondly, the Pressure Map (PM) differs significantly from those in the SLP dataset, failing to accurately capture pressure distribution; it suggests there is high pressure across the entire body with noticeable irregularities. Further examination across other cases confirms that the dataset lacks high quality. However, despite these challenges, this project aims to evaluate the model's performance and assess achievable results.

Two approaches are employed with this dataset. On one hand, the data is directly input into the model to generate predictions. On the other hand, a feature extraction method is implemented where all layers of the pretrained model are frozen except the final one. The model is then fine-tuned on the new data, updating only the weights of the last layer to adapt to the specific characteristics of the new dataset. This transfer learning technique proves valuable in Deep Learning scenarios involving limited dataset sizes and when the original model's architecture does not directly align with the new task requirements.

7.2 Predictions with final model

First of all, predictions are generated for all the data in the UAB dataset, and metrics are extracted. In Figure 11, the prediction with the lowest MSE loss can be observed. It is achieved a MSE Loss of 3.70×10^{-3} and a SSIM of 0.46.

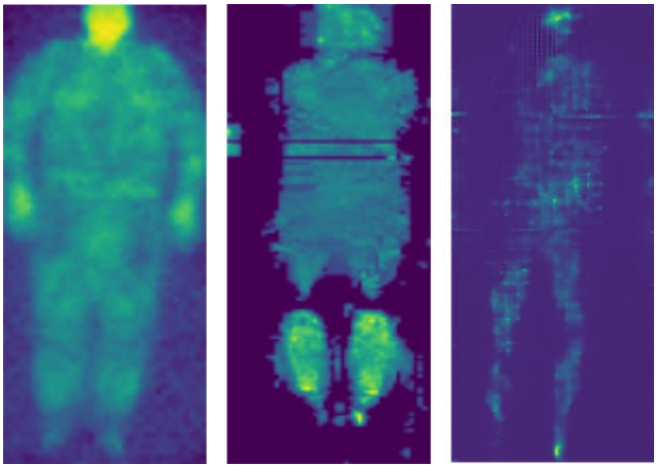


Fig 11: Input, Target and Output of the lowest MSE result

7.3 Feature extraction

As mentioned earlier, the model is trained with the new data, updating only the last convolutional layer. This approach aims to adapt the model to the characteristics of the new dataset. The training process involves using 80% of the data, which totals 540 images, while reserving 126 images for the test set. The model is trained for 100 epochs using MSE loss and a learning rate of 0.02. Figure 12 shows the best case. In this case a MSE Loss of 2.323×10^{-3} and a SSIM of 0.415 is achieved.

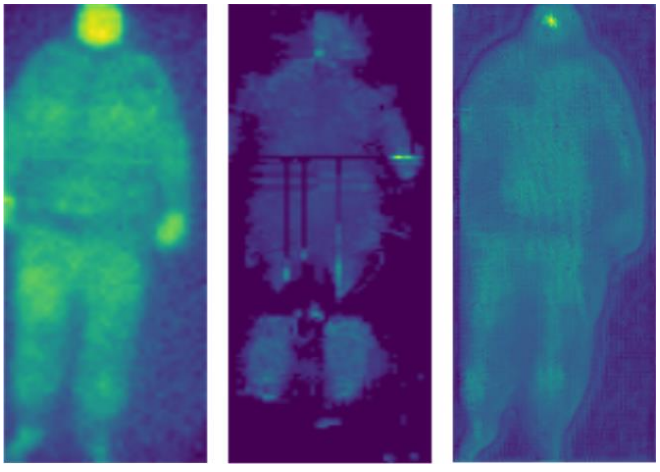


Fig 12: Input, Target and Output of the lowest MSE result using feature extraction

Comparing this with the predictions made using the final model, it is noted that while the MSE Loss obtained through feature extraction is lower, the SSIM is also lower. Figure 12 demonstrates a noticeable difference in the predicted Pressure Maps (PM) compared to previous examples, resulting in a lower SSIM due to dissimilarities.

Upon reviewing both approaches, it is concluded that using the final model trained exclusively with SLP data yields more consistent and reasonable predictions.

8 Conclusions

In this project, a UNET model has been developed to predict pressure maps of human bodies from infrared images. The project began with an extensive preprocessing stage for both the infrared images and the pressure maps. Various loss functions and metrics were defined throughout the project to evaluate the model's performance. The possibility of incorporating patient information vectors into the network was explored, but it was found that this did not significantly improve results. This was attributed to the preprocessing stage already taking into account important physical data of the patient, such as their weight.

Multiple training sessions were conducted with different configurations, and the results were analyzed using various metrics. The final model achieved a mean squared error (MSE) loss of 0.149×10^{-3} with a structural similarity index (SSIM) of 0.9142. Comparing these results with previous works, it was found that similar and optimal outcomes were achieved. Additionally, the model was tested with data from the UAB (Universitat Autònoma de Barcelona), allowing for the study of its generalization and robustness.

In summary, this project successfully addressed the established challenges and achieved the defined objectives. Through meticulous preprocessing, experimentation with various configurations, and thorough evaluation, the developed UNET model demonstrated promising performance in predicting pressure maps from infrared images, laying a solid foundation for future research and applications in this domain. The repository of this project can be found in this link [13].

Acknowledgement

I want to express my sincere gratitude to my thesis supervisor, Vanessa Moreno Font, for her invaluable guidance and support throughout the process. I also want to thank Professor David Castells Rufas from the area of computer architecture and technology. His experience, guidance, and assistance have been fundamental to the completion of this work and overcoming the challenges encountered along the way. Finally, I want to thank Ganyong Mo, a doctoral student in computer vision, for his collaboration and constant exchange of ideas, which have greatly enriched this project. Without the help and cooperation of all of you, this work would not have been possible.

Bibliografia

[1] SLP Dataset for Multimodal In-Bed Pose Estimation, posted by: Sarah Ostadabass, https://web.northeastern.edu/ostadabbas/2019/06/27/multimodal-in-bed-pose-estimation/

[2] Shuangjun Liu, Sarah Ostadabbas, "PressureEye: In-bedContactPressureEstimation viaContact-lessImaging", *Augmented Cognition Lab, Department of Electrical and Computer Engineering, Northeastern University, Boston, MA, USA*, January 2022

[3] Henry M. Clever, Patrick L. Grady, Greg Turk, and Charles C. Kemp, "BodyPressure - Inferring Body Pose and Contact Pressure from a Depth Image". *IEEE transactions on pattern analysis and machine intelligencie*, Vol. 45, No. 1, Janurary 2023.

[4] Scipy documentation, scipy.signal.medfilt2d, https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.medfilt2d.html

[5] Mostafa Wael, "Cook your First U-Net in PyTorch", May 12, 2023, https://towardsdatascience.com/cook-your-first-u-net-in-pytorch-b3297a844cf3

[6] PyTorch Image Segmentation Tutorial with U-NET: everything from scratch baby. "Youtube", uploaded by Aladddin Persson, Feb 2021, https://www.youtube.com/watch?v=IHq1t7NxS8k

[7] Kenneth Leung, 2023, Neural-Network-Architecture-Diagrams, https://github.com/kennethleungty/Neural-Network-Architecture-Diagrams

[8] Pranjal Datta, Sep 2020, All about Structural Similarity Index (SSIM): Theory + Code in PyTorch, https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code-in-pytorch-6551b455541e

[9] Jason Brownlee, Jan 2021, Gentle Introduction to the Adam Optimization Algorithm for Deep Learning, https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

[10] Vrunda Bhattbhatt, "A Step-by-Step Guide to Early Stopping in TensorFlow and PyTorch, Jan 2024, https://medium.com/@vrunda.bhattbhatt/a-step-by-step-guide-to-early-stopping-in-tensorflow-and-pytorch-59c1e3d0e376

[11] Adrian Tam, Using Learning Rate Schedule in PyTorch Training, Apr 2023, https://machinelearningmastery.com/using-learning-rate-schedule-in-pytorch-training/

[12] Arjun Chandrababu, "Mixed Input Data in Pytorch: CNN + MLP", Dec 2021, https://medium.com/@iamarjunchandra/mixed-input-data-in-pytorch-cnn-mlp-8aeff336e8a3

[13] Rito Ghosh, "How To Use Numerical Data with ConvNets", Dec 2020, https://medium.com/learningdeep/how-to-use-numerical-data-with-convnets-4161a628ce46

[14] Gabriel Gausachs, Jun 2024, PM from IR images of the human body in bed Degree Final Project [Python], https://github.com/Gabriel-Gausachs/TFG