Article

Using Deep Learning in Real-Time for Clothing Classification with Connected Thermostats

Adán Medina [1], Juana Isabel Méndez [1], Pedro Ponce [1,*], Therese Peffer [2], Alan Meier [3] and Arturo Molina [1]

[1] School of Engineering and Sciences, Tecnologico de Monterrey, Mexico City 14380, Mexico; adan.mr@tec.mx (A.M.); a01165549@itesm.mx (J.I.M.); armolina@tec.mx (A.M.)
[2] Institute for Energy and Environment, University of California, Berkeley, CA 94720, USA; tpeffer@berkeley.edu
[3] Energy and Efficiency Institute, University of California, Davis, CA 95616, USA; akmeier@ucdavis.edu
* Correspondence: pedro.ponce@tec.mx

Abstract: Thermal comfort is associated with clothing insulation, conveying a level of satisfaction with the thermal surroundings. Besides, clothing insulation is commonly associated with indoor thermal comfort. However, clothing classification in smart homes might save energy when the end-user wears appropriate clothes to save energy and obtain thermal comfort. Furthermore, object detection and classification through Convolutional Neural Networks have increased over the last decade. There are real-time clothing garment classifiers, but these are oriented towards single garment recognition for texture, fabric, shape, or style. Consequently, this paper proposes a CNN model classification for the implementation of these classifiers on cameras. First, the Fashion MNIST was analyzed and compared with the VGG16, Inceptionvv4, TinyYOLOv3, and ResNet18 classification algorithms to determine the best clo classifier. Then, for real-time analysis, a new dataset with 12,000 images was created and analyzed with the YOLOv3 and TinyYOLO. Finally, an Azure Kinect DT was employed to analyze the clo value in real-time. Moreover, real-time analysis can be employed with any other webcam. The model recognizes at least three garments of a clothing ensemble, proving that it identifies more than a single clothing garment. Besides, the model has at least 90% accuracy in the test dataset, ensuring that it can be generalized and is not overfitting.

Keywords: clothing classifier; CNN models; thermal comfort; connected thermostat

1. Introduction

Clothing insulation is commonly associated with indoor thermal comfort. ASHRAE defines clothing insulation as the resistance to sensible heat transfer provided by a clothing ensemble, expressed in units of clo [1]. There are predictive models of clothing insulation that consider outdoor temperature, season, climate, indoor air temperature, indoor operative temperature, relative humidity [2–4]. Rupp et al. [5] evaluated the clothing insulation collected in the ASHRAE database II [6] to predict garment insulation from the indoor air temperature, the season, and building ventilation type. Moreover, Wang et al. [7] proposed a predictive model of clothing insulation for naturally ventilated buildings using the same ASHRAE database II. Gao et al. [8] considered wind direction, posture, and the reduction of clothing insulation due to airspeed to predict thermal comfort.

Alternatively, object detection and classification have been in rapid development for the last 10 years since the famous AlexNet [9] algorithm won the 2012 ImageNet Large Scale Visual Recognition Challenge and started a Convolutional Neural Networks revolution. Hence, Liu et al. [10] used a Convolutional Neural Network (CNN) to recognize an individual's clothes and activity type by capturing thermal videos as inputs. Kalantidis et al. [11] implemented clothing ensemble recognition from a photograph; however, that proposal was not suitable for a real-time solution due to a slow segmentation classification.

There are real-time clothing garments or clothing characteristics classifiers such as the one proposed by Yang and Yu [12]. They used edge detection to obtain information separate from the background and then perform a technique similar to the model proposed by Chao et al. [13], which uses the Histogram of Oriented Objects (HOG) and Support Vector Machines (SVM) to obtain classifications. Yamaguchi et al. [14] focused their research on subjects with single garments instead of a complete ensemble. Furthermore, some CNN approaches used a modified version of the VGG16. Furthermore, some CNN approaches used a modified version of the

VGG16 [15] to orient the garment recognition towards texture, fabric, shape, or style [16–19]. Nevertheless, those approaches did not produce a complete clothing ensemble classification; hence, they only obtained a single clothing garment classification per image.

Due to the increase in dynamic models, adaptive methods that predict clothing properties must understand how an individual adapts to indoor environments. Matsumoto, Iwai, and Ishiguro [20] used a computer vision system and a combination of HOG and SVM to recognize clothing garments. Bouskill and Havenith [21] used a thermal manikin to determine the relationship between clothing insulation and clothing ventilation with different activities known as metabolic rates. They concluded that clothing insulation has less of an effect than the design and fabric of the clothing garment; thus, they recommended analyzing the clothing garments worn in specific places during specific activities to determine the best outfit that avoids colder or warmer thermal sensations.

Moreover, in [20], the authors used an early piece of computer vision hardware from Omron called OKAO Vision to classify objects by proposing a limit that separated two classes, and depending on where the features of new predictions lay, the SVM classified them. However, the SVM was a binary classifier, which meant it only chose between two classes, making it impossible to use this approach as a real-time clothing insulation calculation method. Additionally, the SVM calculated the gradient of each pixel together for every video frame with each computational cost's class. A real-time implementation of clothing recognition is useful for this field to obtain a real-time clo value.

The idea of using computer vision to detect clothing seems expensive when thinking about the implementation of the camera system and the computer needed to process the information and run the solution. However, as cameras are being spread across different uses such as telecare [22–24] or combined with personal assistants such as Alexa [25,26], the concept of cameras being part of the smart home infrastructure needs to be considered. Thus, there would be no need to invest in a camera system and only think about the processing part of the problem.

In [25], the authors proposed using Alexa and a camera to track seniors' moods and emotions to prevent social isolation and depression. In [26], the authors considered Alexa for depression pre-diagnosis and suggested using cameras to track householders. Figure 1 displays the smart home structure. Hence, cameras can track garments. For example, through a smart TV, if possible, camera detection can monitor householder reactions or postures and profile end-users' garments. Thus, this picture shows the integration of household appliances that can help to track householders' daily activities and moods. Moreover, in [26], the authors established for the first time the concept of a gamified smart home to help end-users to save energy without feeling compromised. Besides, previous research had been focused on reducing energy consumption through gamified elements [26–31]. A smart home uses socially connected products [32–36] to profile end-users based on their personality traits, types of gamified user, and energy users to propose tailored interfaces that help them to understand the benefits of saving energy. Moreover, during this research, the authors suggested considering thermal comfort for energy reductions [26,37].

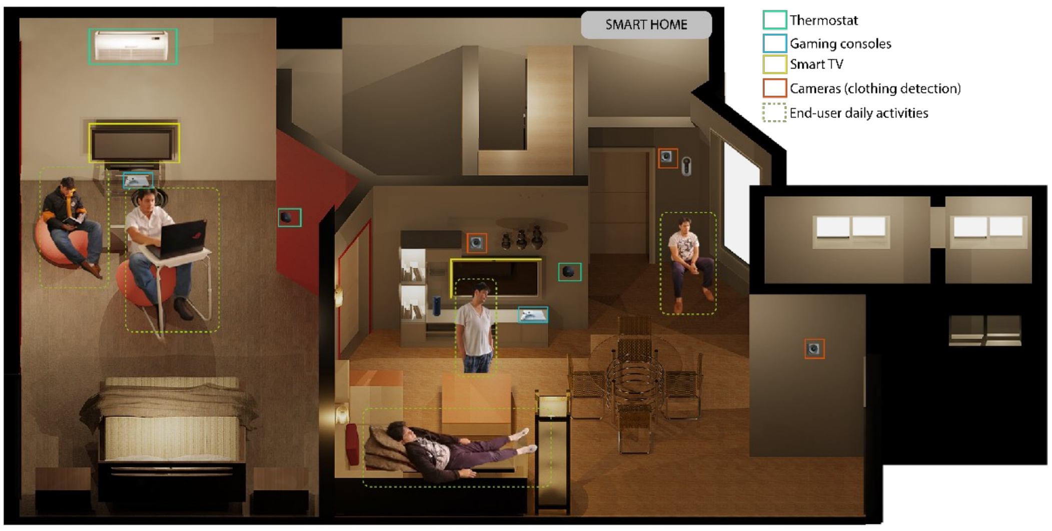


Figure 1. Household devices integration in smart homes with cameras to classify householders' garments.

Therefore, a computer vision system integrated into camera recognition is needed to implement a real-time clothing insulation recognition system to obtain real-time feedback on thermal comfort. Integrating this clothing classifier within the thermostat interface may allow real-time feedback and monitoring to help the end-user to understand how their clothes affect thermal comfort. Besides, increasing the setpoint by 1 °C could save electricity consumption by 6% [38].

Thus, dynamic interfaces could use gamified elements to engage the householder in enjoyable activities while saving energy. There are intrinsic and extrinsic game elements for energy applications provided in the interfaces to help reduce energy [32,37,39]:

- Extrinsic elements: Offers, coupons, bill discounts, challenges, levels, dashboard, statistics, degree of control, points, badges, leaderboard;
- Intrinsic elements: Notifications, messages, tips, energy community, collaboration, control over peers, social comparison, and competition.

1.1. Object Classification Algorithms

A CNN handles multiple dimensions due to the convolutional layers [40]. Hence, there are two types of approaches [41]:

- One-stage: The object detectors produce bounding boxes that contain the detected objects without a region proposal;
- Two-stage: The object detectors carefully review the entire image, leading to a slower process than the one-stage approach but with better accuracy.

Table 1 describes the five CNNs that markedly contributed to the CNN architecture and classifiers.

Table 1. CNN classification algorithms.

CNN	Characteristics	Author
AlexNet [9]	This CNN has eight layers: five convolutional layers connected by max-pooling layers, followed by three fully connected layers. Then, the CNN is divided into two stages, with the feature extraction part done by the convolutional layers and the classification part performed by the fully connected layers. This became the basis for image classifiers.	Alex Krizhevsky
VGG16 [15]	VGG16 consists of convolutional layers stacked on each other. This architecture does not change the size of the kernels in the convolutional layers and keeps it constant in a 3 × 3 value.	Researchers from the Oxford University
GoogleLeNet [42] or Inception	The designers proposed a Convolutional Network with a kernel size of 1 × 1 to reduce the image. Therefore, the CNN significantly reduced the number of parameters needed for the training. This architecture produced better results than the existing algorithms at that moment.	Google
ResNet [43]	This algorithm introduced the residual blocks, which are layers connected in which some weights skipped those convolutional layers. Therefore, deeper networks are implemented to get rid of the degradation problem.	Windows
YOLO [44,45]	YOLO stands for You Only Look Once and is a one-stage algorithm proposed in 2016. This algorithm eliminated the region proposals method of two-stage detector algorithms and instead produced bounding boxes. Thus, the probabilities of the object inside that bounding box belonged to that class. Although this algorithm presents lower accuracy than two-stage object detectors, it can be considered an accurate model.	Joseph Redmon
Tini YOLO [46]	Tiny YOLO is a modified version of YOLOv3 that keeps the algorithm's speed while making it computationally less expensive. Thus, the embedded systems can have the trained model to produce predictions without expensive GPUs.	Joseph Redmon

TensorFlow [47] is an end-to-end open-source platform written in Python and C++ that provides tools and libraries to allow easy implementation of a machine learning application since it provides a tool for the necessary creation, training, deployment, and performance analysis [48]. In addition, it provides Application Programming Interfaces (APIs) which help to create a model with few lines of code. Therefore, the user spends more time focused on the model implementation and its parameters and less time on the coding part of the implementation.

TensorFlow uses data in the form of tensors or arrays of multiple dimensions, also called matrices, and all the operations inside Tensorflow work with these tensors.

Another plus of the Tensorflow package is that it handles data more efficiently and tries to avoid the Graphics Processing Unit (GPU) or Tensor Processing Unit (TPU) waiting for the Central Processing Unit (CPU) to deal with the input data by using its API, called tf.data, to achieve a more efficient importation of the dataset and all the treatment needed so that the GPU/TPU does not suffer from data starvation.

One of the most important APIs contained within the Tensorflow package is Keras [49]. Keras is an open-source deep learning library that was designed to quickly build and train neural network models. It can build these models using the sequential method, which consists of adding layers in turn with the indicated activation function and filter size [48].

Even though there are some object classification models directed towards clothing recognition, most of the proposed algorithms are for fashion industry problems or produce single clothing garment classifications and fail to generalize to other solutions and fail to be able to be implemented in activity recognition or other areas where a real-time clothing ensemble classification may be useful. Hence, this paper proposes a CNN model classification for implementation on real-time devices, such as cameras: the clothing ensemble classifier.

The concrete contributions of this paper are as follows:

- The model recognizes at least three garments of the clothing ensemble, proving that it recognizes more than a single clothing garment;
- The model had at least 90% accuracy in the test dataset, ensuring that it can generalize and it is not overfitting.

Furthermore, the VGG16, Inception, TinyYOLOv3, and ResNet classification algorithms were selected in this study because they are the most basic architectures for image classification. Besides, the previous approaches found in the state of the art of clothing recognition models took as a base architecture the VGG16 architecture [14,18]. Therefore, the aim of this study was to compare the basic architectures to identify which was the best real-time clothing classifier. Furthermore, as the classifier will be used at home, TinyYOLOv3 has a small architecture that can be implemented on embedded systems such as the Raspberry, FPGA, or NVIDIA Jetson Nano.

2. Materials and Methods

Figure 2 displays the methodology used during this research. First, Fashion MNIST was analyzed and compared with the VGG16, Inception, TinyYOLO, and ResNet classification algorithms to determine the object classifier that best suited the clo classification. Then, for the real-time analysis, a new dataset with 12,000 images was created and analyzed with YOLOv3 and TinyYOLO. Since most real-time solutions used the YOLO algorithm, a YOLO model was trained to obtain a real-time clothing garment classifier. Besides, a Tiny YOLO model was tested for the intimacy of the users. Research suggested that Tiny YOLO can be implemented for real-time image detections in constricted environments and implemented into an embedded system. Furthermore, the Tiny YOLO was trained with the recommended weights from another large-scale object detection, segmentation, and captioning dataset known as COCO [50]. Finally, an Azure Kinect DT was employed to analyze the clo values in real-time. Moreover, real-time analysis can be employed with any other webcam.

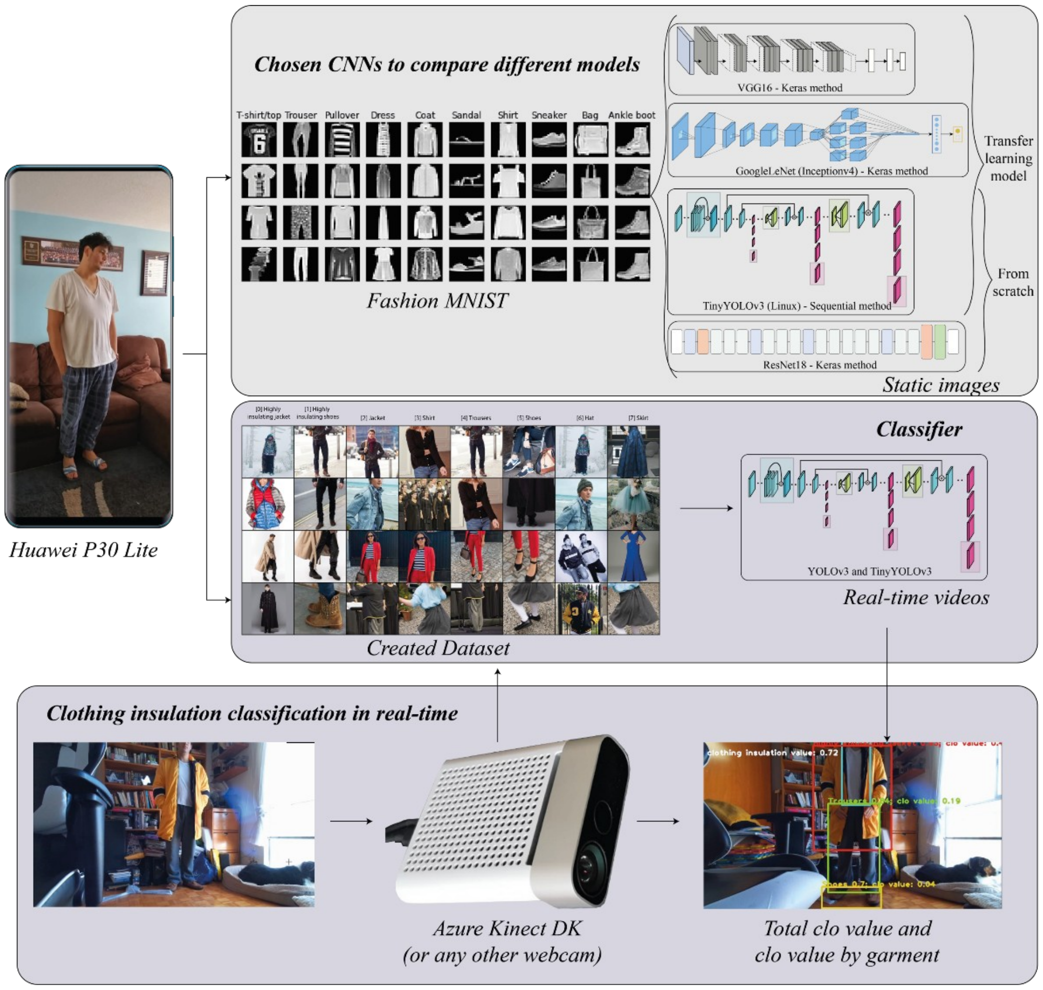


Figure 2. Methodology.

All the tests were performed with a GeForce TX 2080 Ti GPU and an AMD Ryzen 3950 12 core 3.5 GHz processor to avoid any bias during the time measurements. In addition, a Huawei P30 Lite cellphone's camera was used for static images and real-time videos. The recorded images show an individual in a living room walking off camera, changing a garment, walking, and sitting down. The video lasted 24 s.

Furthermore, the current setting did not have more individuals to analyze at the same time; hence, TV series scenes were used to compensate for that lack of individuals and visualize the changes that the model had.

Figure 3 depicts the flowchart used during this research for the entire process for training a neural network.

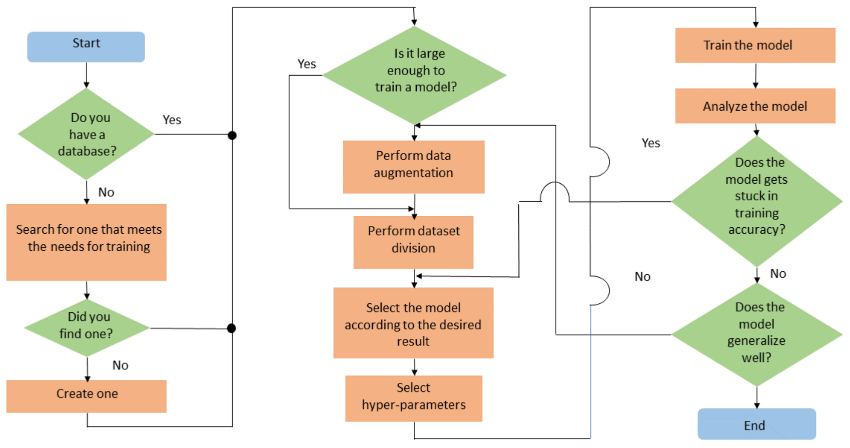

Start
Do you have a database?
Yes
No
Search for one that meets the needs for training
Did you find one?
No
Create one
Is it large enough to train a model?
Yes
Perform data augmentation
Perform dataset division
Select the model according to the desired result
Select hyper-parameters
Train the model
Analyze the model
Yes
Does the model gets stuck in training accuracy?
No
No
Does the model generalize well?
End

Figure 3. NN training flowchart.

2.1. Datasets

Two datasets were analyzed before training the CNN models. The Deep Fashion dataset provided different labeled images grouped into category, texture, fabric, shape, part, and style [16]. The Fashion MNIST dataset [51] provided 70,000 images of clothing garments divided into 60,000 images for training and 10,000 images for testing. The Fashion MNIST dataset was divided into 10 classes: T-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot.

A new dataset consisting of 12,000 images was proposed because of the dataset analysis. Therefore, 2000 images were data augmented to obtain 10,000 additional images, resulting in a total of 12,000 images. These images were divided into sets of 10,800 for training and 1200 for testing. These images were randomly selected from the internet with different backgrounds and different clothing garments worn. The classes were decided based on the premise of keeping the training time at a minimum but having eight different classes to be recognized. Besides, due to hardware and time constraints, only eight labels were selected. Hence, Table 2 presents the eight different classes that were considered. Furthermore, dresses were labeled as skirts due to the similarity of the bottom part of the clothing garment. During the study, there was no access to a computer with a Linux operating system, so Google Colab was used instead to train both networks (YOLO and
TinyYOLOv3) with a custom dataset. However, Google Colab limited the GPU access time, and the 60+ hours needed to train the network translated into several weeks.

Table 2. New dataset labels.

Label	Description
0	Highly insulating jacket
1	Highly insulating shoes
2	Jacket
3	Shirt
4	Trousers
5	Shoes
6	Hat
7	Skirt

The labelImg library was used to label the images because this library allows images and texts to be handled to classify them through bounding boxes. Thus, this classification is compatible with the YOLO format. The advantage of YOLO is that it can detect and classify various objects inside an image, which is perfect for a clothing ensemble classification problem; therefore, this was ideal for the research scope. After labeling the 2000 images, the data augmentation was performed using the clodsa library, as the train and test files were created in the YOLO format. This library was used to perform image transformations on the labeled dataset by keeping the bounding boxes in the correct place. Thus, the augmented dataset increased to 12,000 images.

2.2. CNN Algorithms

Figure 4 depicts the four CNN algorithms used during this research. The VGG16 and Inceptionv4 considered preloaded weights as the Keras application class [15,48]. The ResNet18 [43,48] was built from scratch, and the TinyYOLOv3 [46] had no preloaded weights and was built from scratch. Thus, the ResNet18 allowed us to make a comparison with the Tiny YOLO model. Moreover, the Tiny YOLO model considered preloaded weights. It was trained with Linux commands to perform a comparison with the independent images.

Furthermore, the validation and training accuracy and the confusion matrices were plotted to compare each CNN. Besides, five images were tested to obtain the clo value in real-time. The model detected the eight different classes and displayed the total clo value and the probability percentage of belonging to the class they examined.

The Tiny YOLO considered pre-trained weights because the other CNN models used these feature extractors as their method to obtain weights. Thus, the Tiny YOLO performed the same tests on the same images and compared them with the other CNN models. Furthermore, the comparison made it possible to visualize any difference with the model created from scratch. Finally, the last model was trained in Google Colab and operated with the new custom dataset of 12,000 images. These 12,000 images came from the original image dataset of 2000 images gathered from the internet with no particular size. These images were labeled with the YOLO format; then, data augmentation techniques were used to add rotation, hue changes, contrast changes, horizontal flip, Gaussian noise, and gamma color correction to enlarge the dataset and cover more areas where the camera and lighting settings may affect the effectiveness of the model.

The threshold refers to the models' confidence percentage that a detected object belongs to a class. For instance, a threshold value of 0.5 generated bounding boxes around objects that have over 50% probability of belonging to the predicted classes. Thus, this project used an initial threshold of 0.4 and lowered it to 0.2 for the video test. For the images, the threshold was set to 0.1 to study all the classifications the model makes.

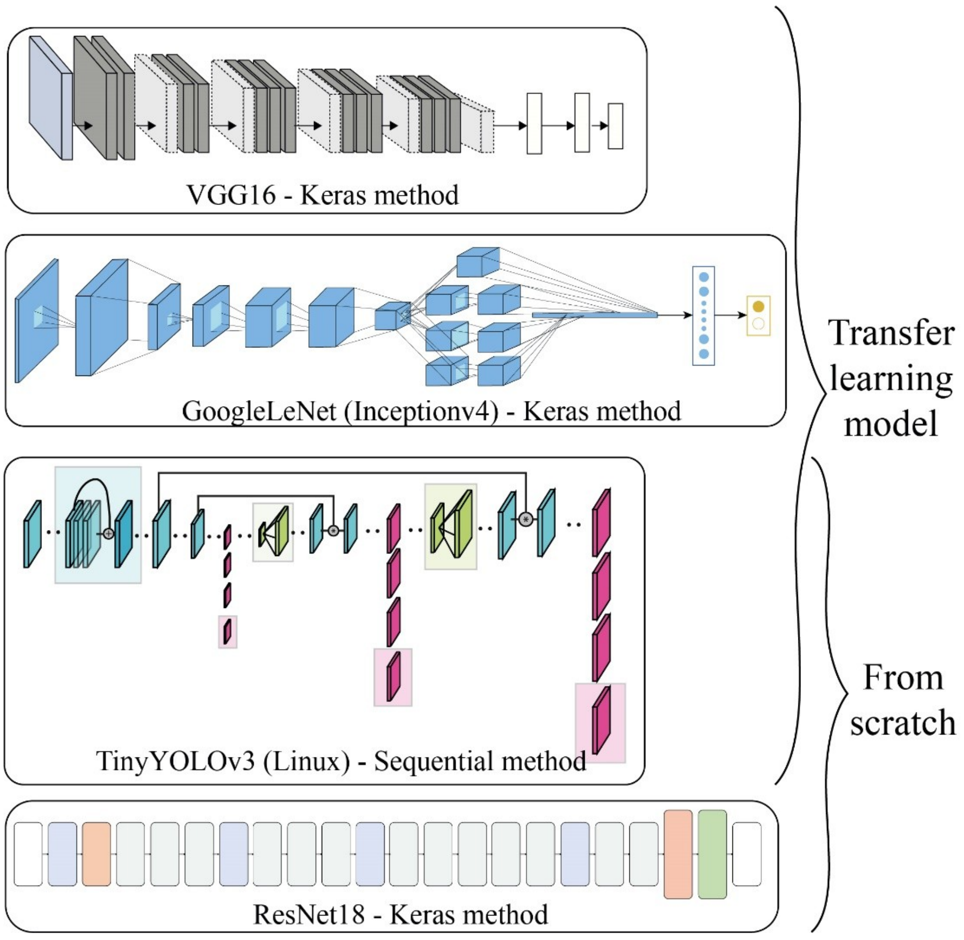

VGG16 - Keras method
GoogleLeNet (Inceptionv4) - Keras method
TinyYOLOv3 (Linux) - Sequential method
ResNet18 - Keras method
Transfer learning model
From scratch

Figure 4. CNN model architecture for VGG16, Inceptionv4, TinyYolov3, and ResNet18.

2.3. Training the Models

The training was conducted using 100 epochs, a batch size of 128, the Adam optimizer, and a constant learning rate of 0.01. A change to a test of 0.05 was tried with no significant change. Besides, due to time and hardware constraints, changing the parameters using Google Colab would take several weeks. Thus, no further changes were performed for this hyperparameter. However, these parameters could be tested in future work.

The compile method configured the network, whereas the fit method was used for the model training. Therefore, the training dataset was the input parameter. Besides, the number of batches, epochs, and callbacks were chosen. The batches were divisions of the dataset used to train on a random portion instead of the whole dataset to avoid the failure of the computer due to not having enough Random Access Memory (RAM) to produce the result.

Google Colab was used for the YOLO training because a Linux command was needed to create the required environment and train the model. However, the main drawback was the time limit. Google Colab allows access to its GPU for 4 h; then, it is necessary to wait for 18 h to gain access again for another 4 h. Hence, the Tiny YOLO model training took 90 h, of which 60 h corresponded to the training time.

Currently, there has been no comparison between Tiny YOLOS's feature extraction architecture with the other common architectures. Consequently, during this research, the comparison was performed against the other image classifiers. Furthermore, neither YOLO nor Tiny YOLO was previously implemented on Keras. Hence, the sequential method was built to compare the different CNN models with Tiny YOLO. Moreover, this model was built from scratch using the same activation functions and the number of filters, sizes, and strides to keep the model as close to the original as possible.

As for the VGG16, Inceptionv4, and ResNet18 models, Keras included a method to call on these models and used them as feature extractors to add a few dense layers and differentiate the classifications. Thus, this method was used to compare the models.

2.4. Comparing the Models

The accuracy results were obtained from the training models and were plotted against the epochs to see how the models changed their performance along with the training and if there was any overfitting present, as well as to recognize if early stopping should have been used to avoid overfitting. This accuracy corresponds to the best accuracy of the validation data and the corresponding best accuracy on the training data for that same epoch.

Additionally, confusion matrices were built using the network's predictions compared to the actual values with the test dataset to analyze if the accuracy was true. A model that classified everything as a negative except a few positive classifications correctly can have an accuracy value of over 90%; nonetheless, this model would be useless. Furthermore, the metrics for model evaluation used during this study were the precision, recall, and F1 score.

2.5. Study Case: Clothing Insulation Real-Time Analysis Applied on Thermostats

Once the CNNs were trained, a real-time clothing recognition approach that can be implemented at home was proposed. This real-time recognition was oriented to infer clothing insulation based on the clo values presented in Table 3. Figure 5 depicts the flow chart considered during the proposed solution to obtain a real-time implementation of clothing recognition.

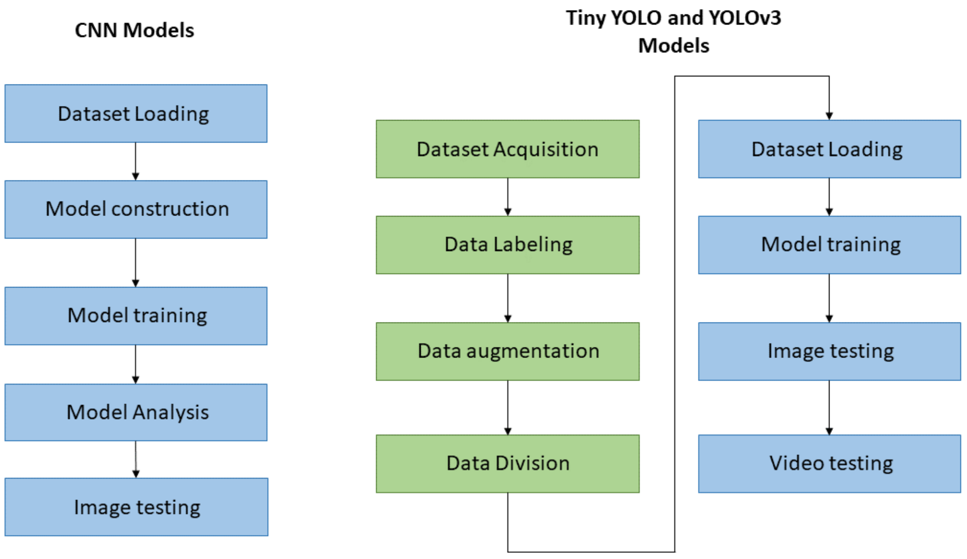


Figure 5. Flowchart for the image processing and Deep Neural Network classification for real-time implementation of clothing recognition.

Thus, the process used in this study was to select the dataset to be implemented in the model training, label the images to add more precision, and provide information about the people wearing the clothes and background. Clothing affects factors that involve the heat transfer between the human body and the ambient environment; besides, clothing insulation affects thermal comfort because the difference in the value of this factor can change the perception of the ambient environment's temperature.

For this reason, both human-centered and building-centered thermal comfort calculations consider clothing insulation as a factor for the overall thermal comfort range. However, a thermal calculation method has been overlooked due to the difficulty of detecting and classifying every clothing garment that a user is wearing.

Table 3. Clothing insulation values considered for the classes [52].

Label	Garment	(clo) [1]	(m^2 C/W)
0	Highly insulating jacket, multicomponent	0.40	0.062
1	Highly insulating shoes, boots	0.10	0.016
2	Jacket, no buttons	0.26	0.040
3	T-Shirt	0.09	0.010
4	Trousers (straight, fitted)	0.19	0.029
5	Shoes [1]	0.04	0.006
6	Warm winter cap	0.03	0.00
7	A-Line, knee-length	0.15	0.023

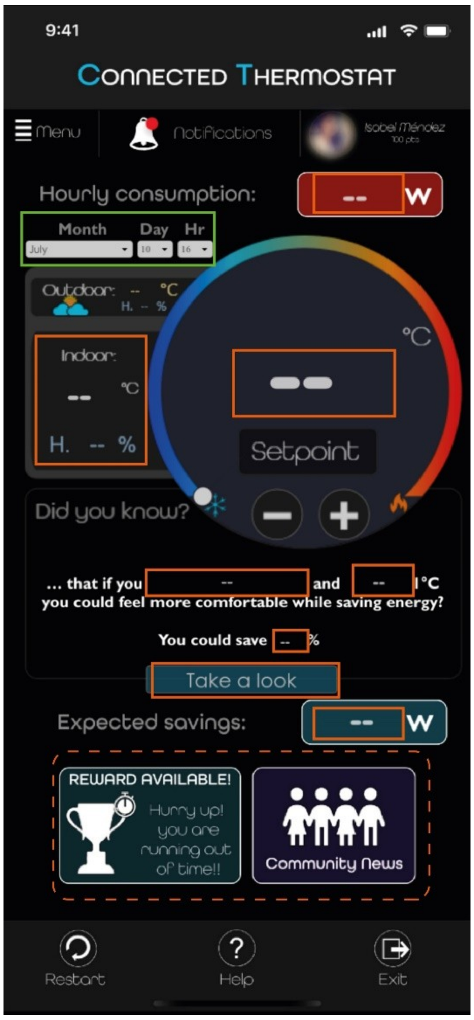
[1] 1 clo = 0.155 m^2 C/W.

Then, a home located at Concord, California, was energy simulated to measure the impact of increasing or decreasing the temperature by 1 °C at the HVAC setpoint. A change of 1 °C can save 6% of electricity [38]. Other elements fed into the energy model were a weather file from Concordia, the construction materials, the home schedule, and loads. The energy model simulation used LadybugTools v1.4.0 from Rhinoceros + Grasshopper. The living room zone was analyzed to obtain the HVAC consumption and calculate the PMV/PPD to determine if the householder was comfortable. The parameters considered were a metabolic rate of 1.0 and a dynamic clo value based on Table 3.

Then, a dynamic interface was proposed based on the energy model results. This interface was built in MATLAB/Simulink V.R2021a. This interface determined, in an interactive and ludic manner, how to save energy by modifying the setpoint and suggesting appropriate types of clothes. Figure 6 displays the input values inside a green box (interface, the month, day, and hour); the output values were the hourly consumption in Watts, the outdoor and indoor temperature, the relative humidity, the setpoint, and the expected savings. In the "Did you know?" section, a message was displayed, and based on the possible energy savings and thermal comfort, three actions were displayed:

1. Wear lightweight clothes;
2. Wear the same clothes;
3. Wear warmer clothes.

The "Take a look" button showed the householder the potential savings achieved by performing those actions. The "Reward available" and "Community news" elements belong to a gamification structure. These buttons are displayed this way because gamification theory suggests that promoting intrinsic motivations and extrinsic motivations in real activities can achieve specific goals, such as energy reduction [28,32,37,39].



... that if you -- and -- °C you could feel more comfortable while saving energy?

You could save -- %

Figure 6. Thermostat interface proposal.

3. Results

Although the proposal was to use both datasets to compare results, only Fashion MNIST was considered because the Deep Fashion dataset was encrypted and required a password to decompress the dataset files. Therefore, an e-mail was sent to the authors, but we never received a response or the required password. Thus, a new dataset was created.

3.1. Datasets Treatments

Figure 7 depicts the dataset observation and the divisions considered for training validation and testing stages with the Fashion MNIST dataset. The image shows that the dataset had 10 different classes with the grayscale format and was printed into the NHWC format. Figure 8 shows that the distribution showed no significant difference for the training data and validation data. That means that the model was not biased.

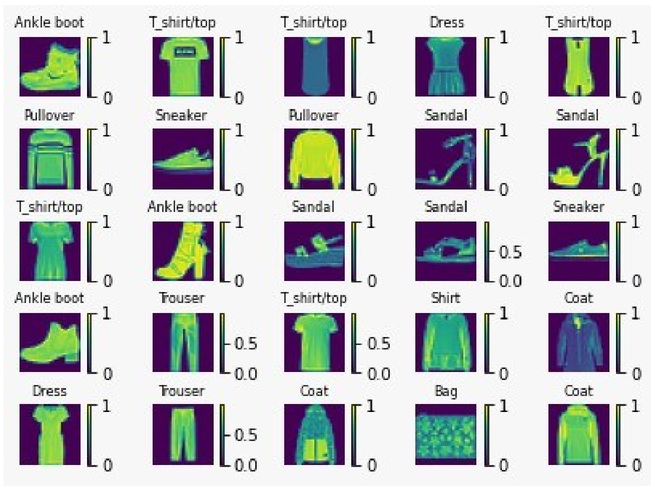


Figure 7. Fashion MNIST observation.

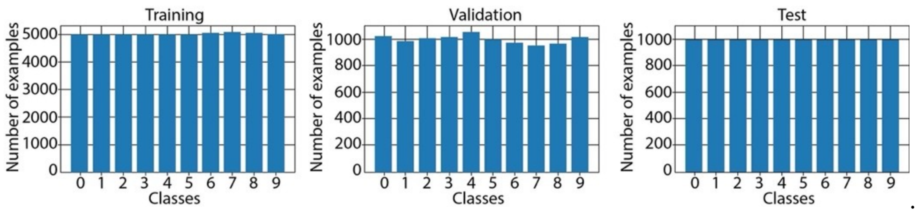
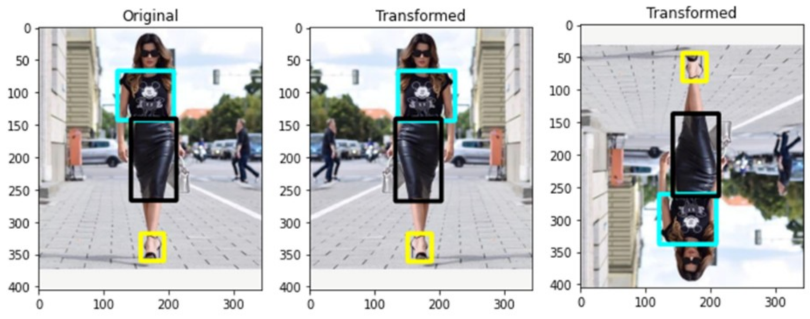


Figure 8. Fashion MNIST division.

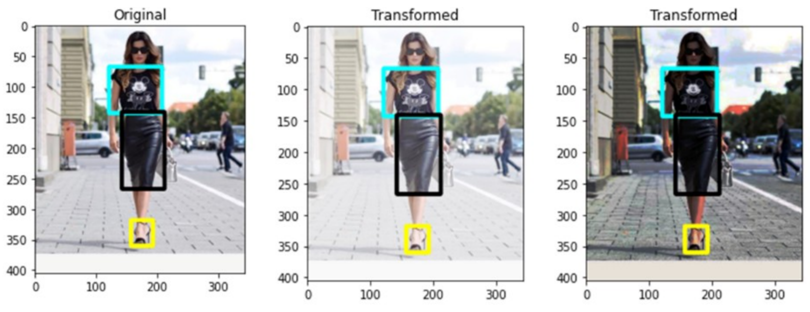
In addition, the data augmentation process used the clodsa package. Figure 9 shows an example of these transformations. Therefore, the results of each transformation allowed the labeled bounding boxes to keep their place without affecting the training.

Figure 9a shows the flipping transformations. They cover different postures of the people; the vertical flip considers some individuals that prefer to lay down with their feet up, for instance, to alleviate feet pain. Figure 9b displays the hue and the contrast transformations to cover different lightning environments and possible impediments for a camera. Figure 9c represents the blurring and histogram transformation. They were selected to cover the difference between the image resolution taken from cameras with fewer megapixels or lower resolution than the ones used for training.

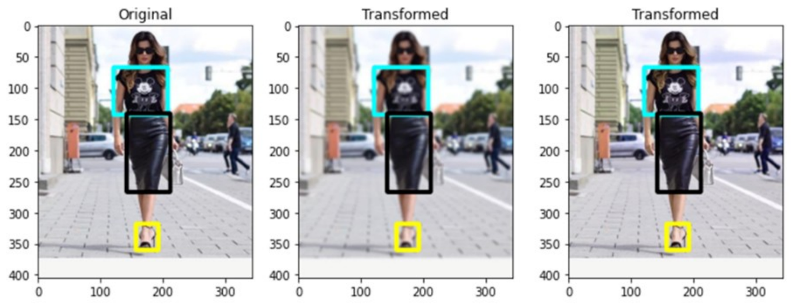
There were some differences in the number of examples containing people wearing jackets, shirts, and trousers because it was relevant to discern between a highly insulating jacket and a regular jacket. Any shoe that covered the ankle was labeled as a highly insulating shoe. Furthermore, no differentiation for sandals was made. Every dress was labeled as a skirt due to its similarity with the skirt's shape. Besides, the objective was to have fewer classes to have less training time, and thus we prioritized recognizing several parts of a clothing ensemble such as shirts, jackets, shoes, and skirts that are more common and have different clothing insulation values.



(**a**)



(**b**)



(**c**)

Figure 9. Data augmentation process: (**a**) horizontal and vertical flip; (**b**) hue and contrast transformation; (**c**) blur and histogram transformation.

3.2. Object Classifiers Comparison

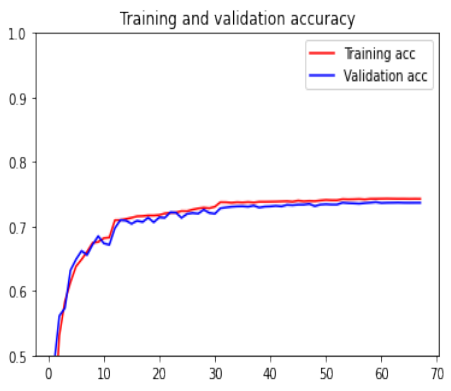
The validation and training accuracy graphs for all models are shown in the following images, where Figure 10a is from the VGG16 network, Figure 10b is from the inception model, Figure 10c is from the ResNet34 network, and Figure 10d is from the Tiny YOLO network.

Figure 10a shows that the reached accuracy was below 0.8. Although this model used preloaded weights, it was not as accurate as the other models. This architecture was the basis for some of the proposed solutions for clothing recognition found in the literature reviews. Therefore, it was relevant to inspect the performance of this algorithm. Even though there was no difference between the accuracy from training and the validation, it had a low score in the accuracy metric compared with the other models.

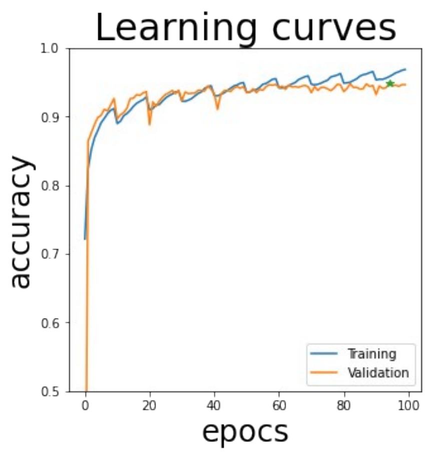
The accuracy graph of the Inception model (Figure 10b) shows that the Inception model fares better in the accuracy metrics when compared with the VGG16 model but failed to reach a stable point within the 100 epochs. Therefore, this model required more epochs. Although, there was little difference between the validation and training accuracy, the ResNet18 and Tiny YOLOv3 models had better scores in both datasets.

Figure 10c reveals that the ResNet algorithm reached a perfect training accuracy, but the validation accuracy was barely above 90%. Hence, overfitting needs to be considered, and implementations of dropout can improve the model. Moreover, early stopping can be considered because the best accuracy for the validation was in the first epochs.

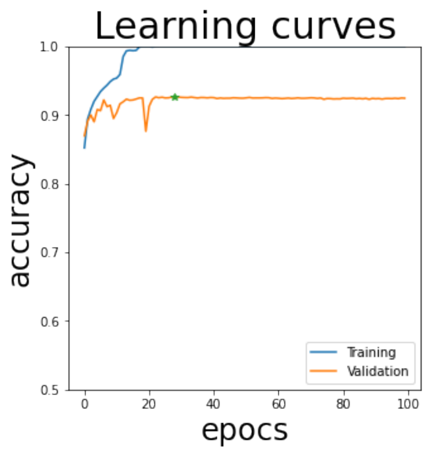
Figure 10d shows that the Tiny YOLO made from scratch had a good accuracy but that there was overfitting since the top accuracy score was reached in the first epochs and the difference between training and validation accuracy was greater than 5%. Therefore, a dropout layer with a 50% drop rate was implemented at the middle of the hidden layers.



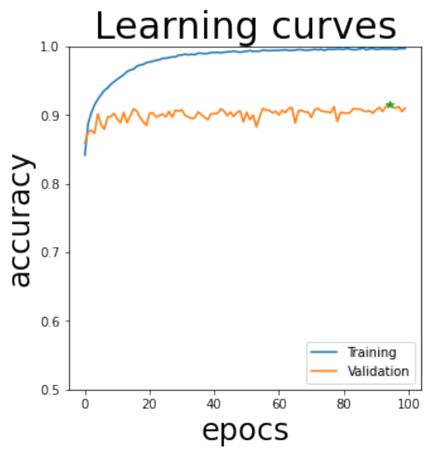
(**a**)



(**b**)



(**c**)



(**d**)

Figure 10. Validation and train accuracy graphs: (**a**) VGG16 accuracy; (**b**) Inception accuracy; (**c**) ResNet accuracy; (**d**) Tiny YOLO accuracy.

Figure 11 presents the result of the dropout layer implementation, showing that there was no discerning difference between the results obtained with and without dropout. Therefore, for this implementation, the difference in accuracy scores was not enough to consider dropout, and the test dataset results needed to be analyzed. Besides, this may also indicate the need for a bigger dataset.

As the accuracy may be a misleading metric, a confusion matrix was employed to make a more complete comparison.

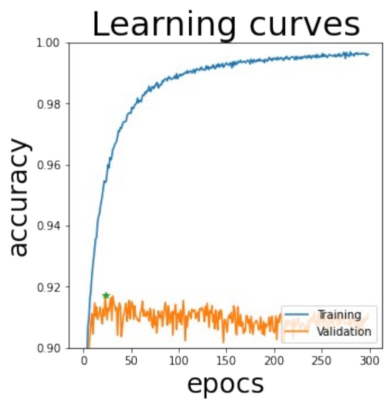


Figure 11. Tiny YOLO with 50% dropout.

Figure 12 depicts the confusion matrices. The model seems to have problems detecting the T-shirt/top, pullover, coat, and shirt classes. The shirt class had more errors because the model misclassified clothing items as a shirt. Hence, this model was sensitive towards the shirt class. Moreover, all the models presented this problem, because it was difficult to separate the shirt class from the T-shirt/top class and some of the coat class examples.

The confusion matrices show that even though ResNet and Tiny YOLO seemed to have better results for the accuracy metric than the Inception model, Inception seemed to perform better in the test dataset. So, to finish this comparison, we consider the numeric values side by side to be able to have a better look at the differences between the models.

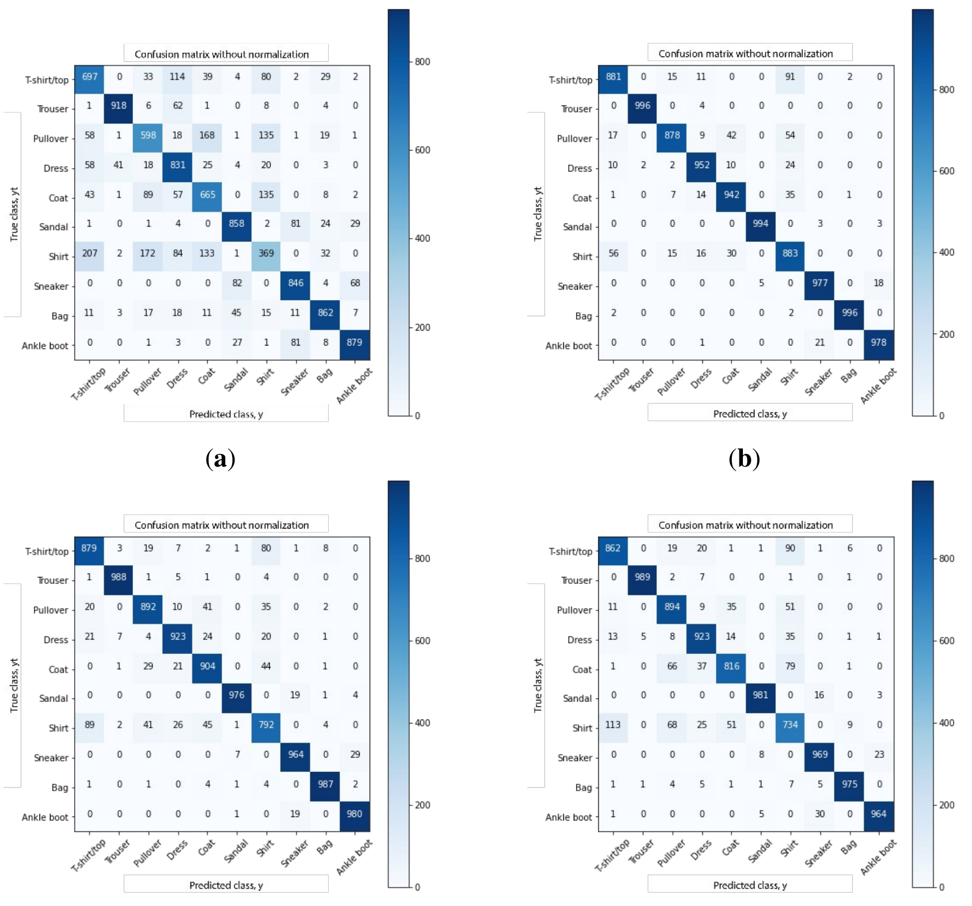


Figure 12. Validation and train accuracy graphs: (**a**) VGG16 accuracy; (**b**) Inception accuracy; (**c**) ResNet accuracy; (**d**) Tiny YOLO accuracy.

Table 4 shows that Tiny YOLO and ResNet18 performed better in the training and validation stages than the other models. The testing accuracy was below that of the Inception model, but this model extracted better features, as confirmed by the recall value.

Therefore, the Inception model was the best model in terms of recognizing clothing garments, but it used preloaded weights. Hence, for real-world implementation, a trained Tiny YOLO was created using Linux commands and used preloaded weights to make a fair comparison. Nevertheless, these commands did not offer a way to see the accuracy in the different datasets used for training, validation, and accuracy determination. Therefore, the testing images were used to compare the models. These images were considered due to the complexity of the postures or objects in front of the individuals.

Table 4. Comparison of models.

Model	Training Accuracy (%)	Validation Accuracy (%)	Test Precision (%)	Test Recall (%)	F1 Score
VGG16	76.16	75.17	75.85	75.85	75.85
Inception	96.0	94.76	95.18	95.07	95.12
ResNet	99.9	92.71	95.46	92.85	94.14
TinyYOLOs	99.48	91.11	91.07	91.10	91.08

A test on five different images that were not part of the datasets was used to test the real implementation of the CNN models since the objective was to train a CNN model with a high accuracy rate in a training dataset and implement it in a real-world situation, where external factors such as noise or light could affect the models' performance.

Figures 11 depicts five images with different postures and garments. Figure 13a shows an individual with a seated posture and lighter garments, Figure 13b shows an individual with reclined posture with a jacket, Figure 13c shows a reading posture with a highly insulated jacket. Figure 13d shows a model in a standing posture with sandals and lighter garments. Figure 13e shows an individual in a writing posture with lighter garments.

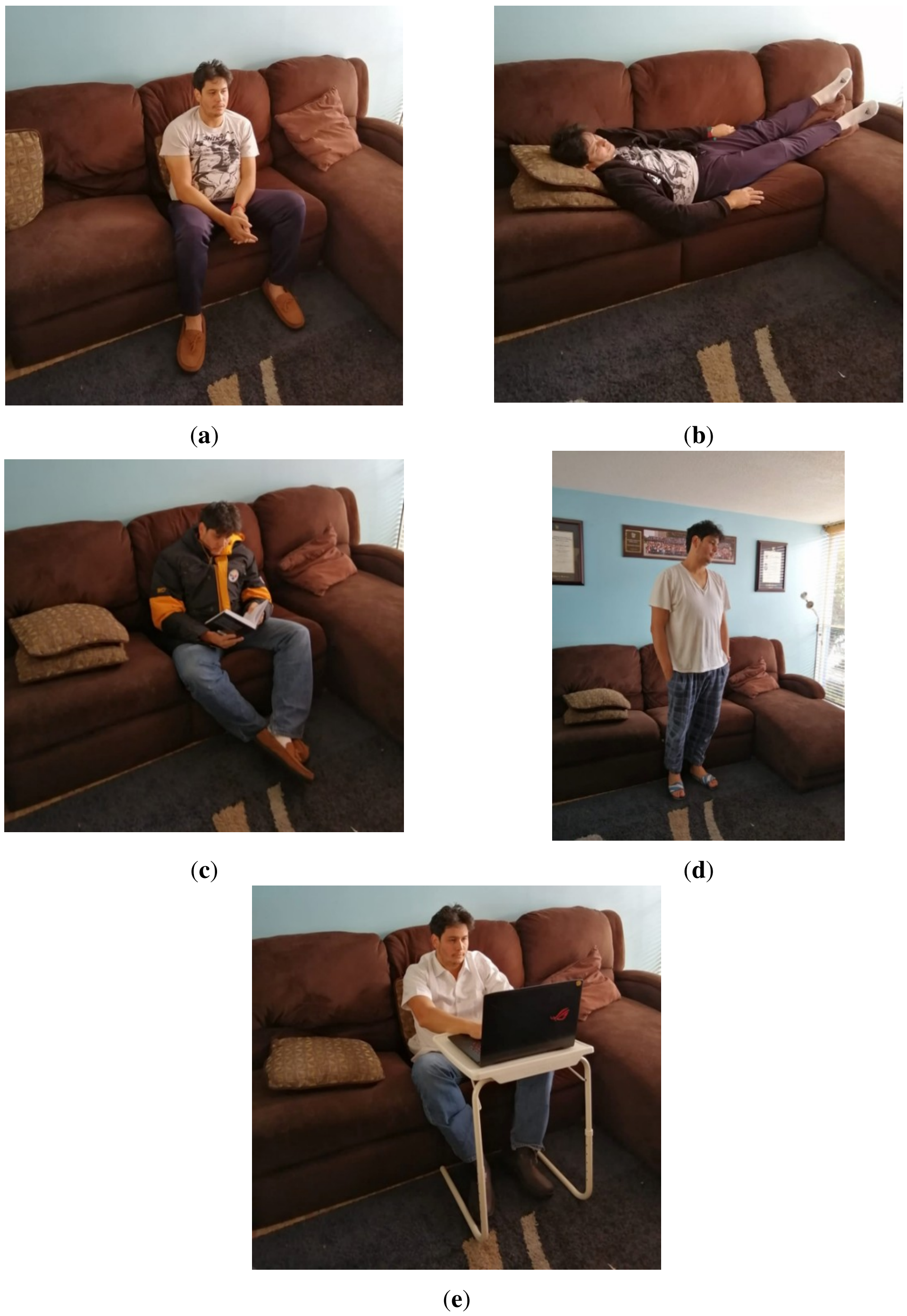


Figure 13. Test photos: (**a**) Test photo 1: seated posture; (**b**) Test photo 2: reclined posture; (**c**) Test photo 3: reading posture; (**d**) Test photo 4: standing posture; (**e**) Test photo 5: writing posture.

Table 5 shows the predicted classes, separating between the top choice for the model and the other possible classes, according to how close the probabilities for the top class were, considering a threshold of 10%. The final column has the time in milliseconds it took for the model to produce the classification. Tiny YOLO from scratch (Tiny YOLOs), Tiny YOLO from Linux (Tiny YOLOl), and the Inception model produced more than one classification. Nonetheless, they had problems differentiating between the T-shirt/top, shirt, coat, and pullover classes.

Moreover, these three models managed with these images to produce multiple classes of classification in most images. Unfortunately, none of them were consistent, possibly due to the lack of additional information from the Fashion MNIST dataset. Besides, these models can be used for clothing ensemble recognition. Nevertheless, they had problems making the correct classifications; thus, bounding boxes were required.

Furthermore, real-time detection algorithms considered the YOLO or Tiny YOLO architecture. The Inception algorithm could present problems for real-time implementation due to the average recognition time. Therefore, the Tiny YOLO model was considered since it tested the possibility of obtain a garment ensemble classifier by using a dataset with more information.

Table 5. Testing on independent images.

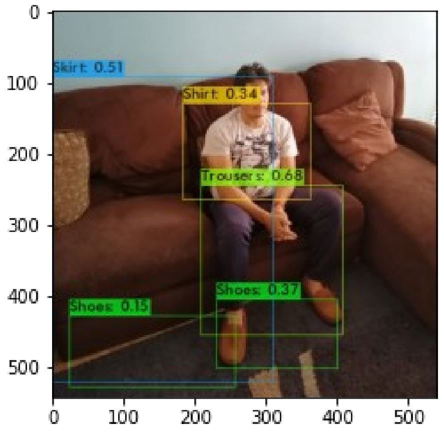
Model	Top Class Probability	Other Classes	Time
	Test photo 1		
TinyYOLOs	Bag	Shirt, Pullover	53
VGG16	T-shirt/top		60
Inception	Shirt	Pullover, Trouser	32
ResNet	Shirt		50
	Test photo 2		
TinyYOLOs	Bag	T-shirt/top, Shirt	57
VGG16	T-shirt/top		68
Inception	Shirt	Pullover	46
ResNet	Shirt		47
	Test photo 3		
TinyYOLOs	Shirt	Bag, Pullover	48
VGG16	Bag		65
Inception	Shirt	Pullover	57
ResNet	Shirt		48
TinyYOLOl	Coat	Pullover	
	Test photo 4		
TinyYOLOs	Bag	Trouser, Shirt	53
VGG16	Ankle Boot		68
Inception	Shirt	Pullover	48
ResNet	Shirt		47
TinyYOLOl	Pullover		
	Test photo 5		
TinyYOLOs	Bag	Shirt, Pullover	52
VGG16	T-shirt/top		72
Inception	Shirt	Pullover, Trouser	50
ResNet	Shirt		40
TinyYOLOl	Coat	Pullover	

3.3. Tiny YOLO and YOLO Using Transfer Learning

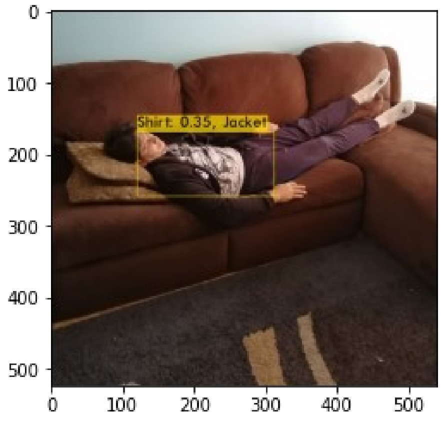
Figure 14 depicts the labeled dataset with bounding boxes for the YOLO and Tiny YOLO models using a threshold of 0.1; then, the threshold was increased for the YOLO model to 0.4 and 0.5.

Figure 14a, shows that the sofa was misclassified as a skirt with a 51% probability. Figure 14b shows that the model only classified the shirt and jacket. As these figures show, we found that a threshold of 0.1 was labeling the sofa and floor, and thus the threshold was increased to 0.4. Hence, Figure 14c shows that the model recognized the jacket, trousers, and shoes but misclassified the coach. Moreover, Figure 14d shows that the model recognized three garments; thus, another test was made by increasing the threshold up to 0.5. Therefore, as shown in Figure 14e,f, the model recognized two clothing garments. This threshold was needed because this algorithm proposes the classification and the bounding boxes for recognized objects. Furthermore, this threshold value was considered to avoid detected objects that were not contained in any of the classes, such as the sofa or the floor.

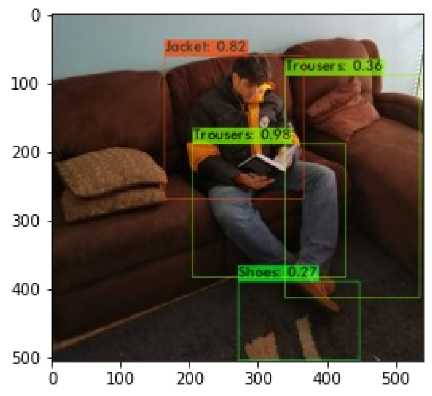
The Tiny YOLO model classified the garments with a threshold of 0.1. Figure 14g shows that the model properly classified all the clothes. Figure 14h shows that the model misclassified the trousers. Figure 14i shows that the model failed in classifying the garments except for the trousers. Figure 14j shows that the model misclassified the sofa as a skirt and trousers. Figure 14k shows that the model wrongly classified the laptop as a shirt.



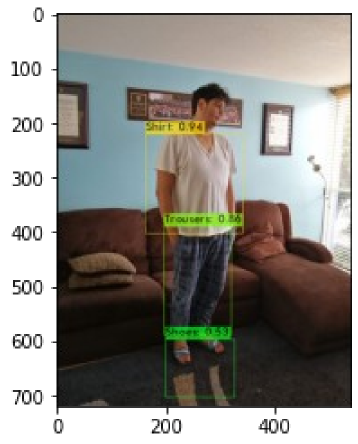
(**a**)



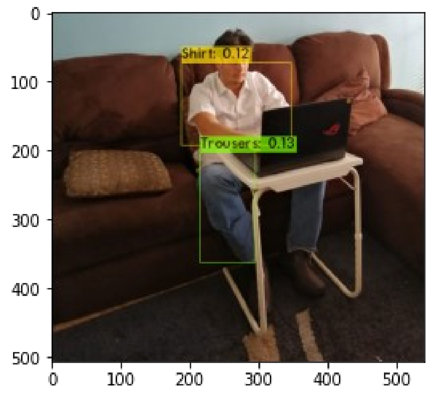
(**b**)



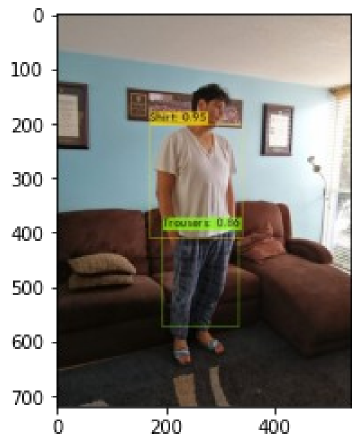
(**c**)



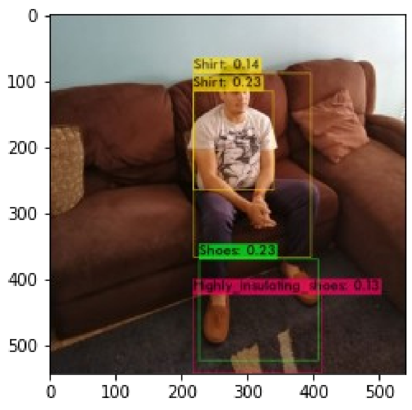
(**d**)



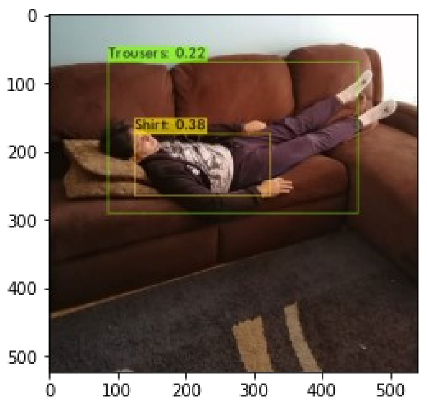
(**e**)



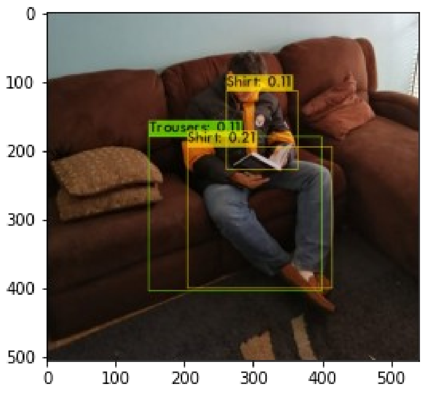
(**f**)



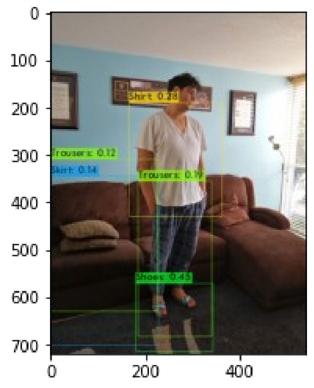
(**g**)



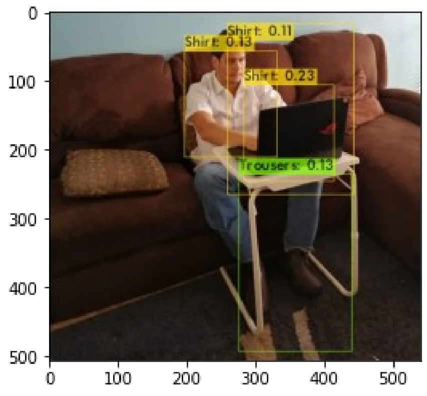
(**h**)



(**i**)



(**j**)



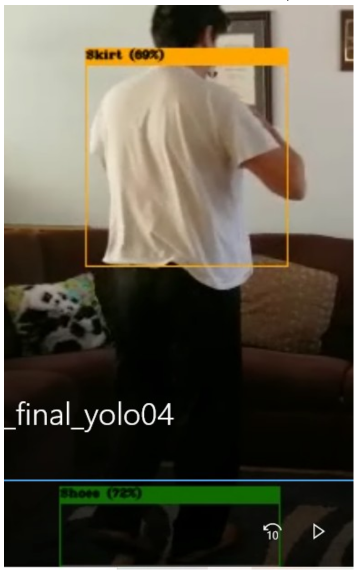
(**k**)

Figure 14. Bounding boxes with a 0.1 threshold: (**a**) YOLO results for test photo 1; (**b**) YOLO results for test photo 2. YOLO results: (**c**) test photo 3 with 0.4 threshold; (**d**) test photo 4 with 0.4 threshold; (**e**) test photo 5 with 0.4 threshold; (**f**) test photo 4 with 0.5 threshold. Tiny YOLO results with 0.1 threshold: (**g**) test photo 1; (**h**) test photo 2; (**i**) test photo 3; (**j**) test photo 4; (**k**) test photo 5.

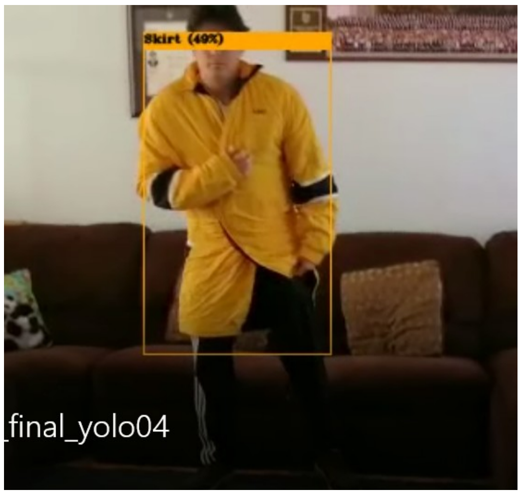
Figure 15 depicts the best and worst video results for the YOLO and Tiny YOLO models with 0.4 thresholds. Screenshots were taken to produce the results and show them in this paper. Figure 15a,b show that the model misclassified the shirt as a skirt because the model understood that this type of shirt seemed more like a skirt. However, Figure 15a shows that the model reflected the best classification for the YOLO model. In Figure 15c, the Tiny YOLO model classified correctly the highly insulated jacket. Nevertheless, Figure 15d shows that the model misclassified the sofa as trousers. Therefore, the threshold was decreased up to 0.2 to review if there were more classifications that the model obtained for multiple garment detections. Since the threshold was lower, there were more resulting images. Figures 15e–h depict the worst classifications, whereas Figures 15i–l show the best classifications.

Figure 15e shows that the model misclassified the sofa as trousers. Figure 15f shows that the model misclassified half of the scene as a highly insulated jacket, some books as shoes, and the shirt as a skirt. Figure 15g shows that the model did not recognize the individual's garments and misclassified the sofa as trousers and highly insulated shoes. Figure 15h shows that the model misclassified the sofa as a shirt, the floor, the shirt, and the shoes as trousers.

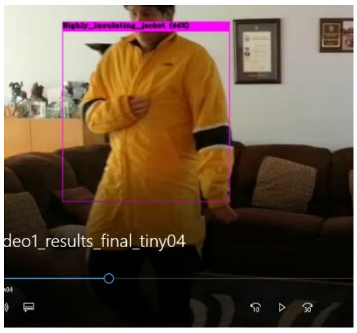
Figure 15i shows that the model classified the garment as a highly insulated jacket and, due to the shape, also as a skirt. Figure 15j shows that the model correctly classified the shows and the shirt, and even suggested that it could be a skirt; nevertheless, it misclassified the trousers as highly insulated shoes or a skirt. Figure 15k shows that the model correctly classified the clothing as a highly insulated jacket but misclassified it as a trouser. Figure 15l shows that the model classified the shirt and trousers; however, the model considered the trousers to include the sofa.



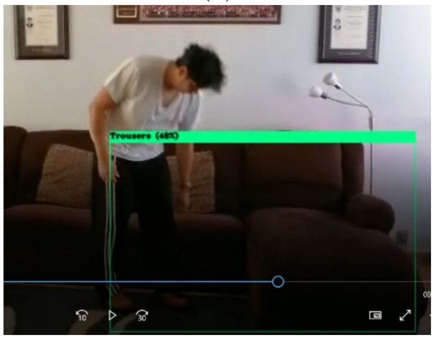
(**a**)



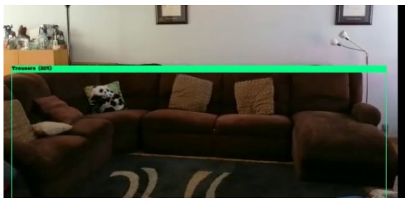
(**b**)



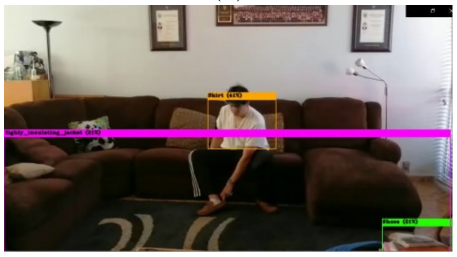
(**c**)



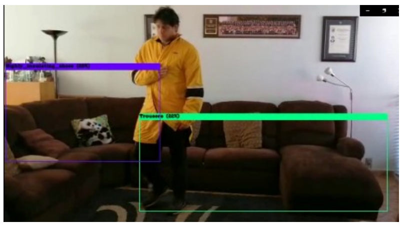
(**d**)



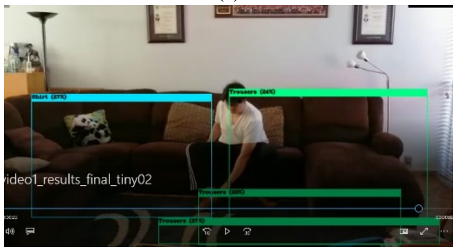
(**e**)



(**f**)



(**g**)



(**h**)

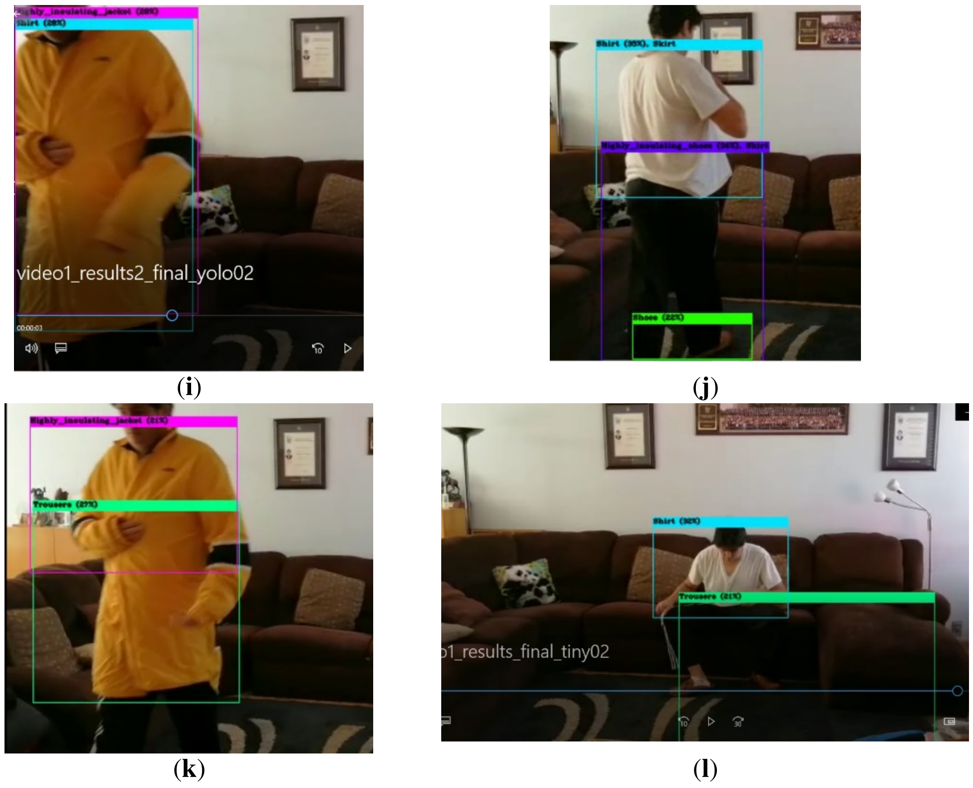


Figure 15. Video results with 0.4 thresholds: (**a**) YOLO's best classification; (**b**) YOLO's worst classification; (**c**) Tiny YOLO's best classification; (**d**) TinyYOLO's best classification. Worst video results with 0.2 thresholds: (**e**) YOLO with sofa view; (**f**) YOLO with a seated individual; (**g**) Tiny YOLO with highly insulated jacket; (**h**) Tiny YOLO with a seated individual. Best video results with 0.2 thresholds: (**i**) YOLO with highly insulated jacket; (**j**) YOLO with a shirt; (**k**) Tiny YOLO with highly insulated jacket; (**l**) Tiny YOLO with a seated individual.

3.4. Study Case: Clothing Insulation Real-Time Analysis Applied on Thermostats

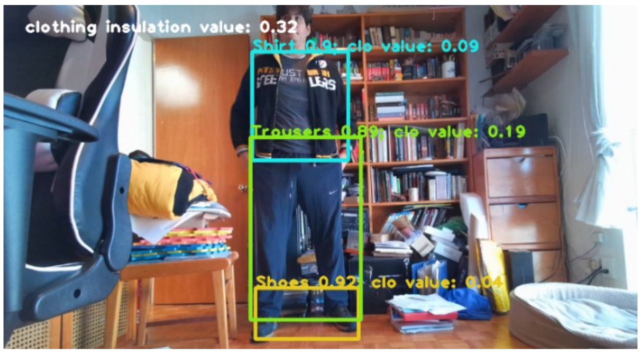
The clothing insulation values are shown in Table 3. Moreover, since the previous video testing did not produce proper classifications, multiple users, garments, and posture were tested on a TV show. Nevertheless, to avoid any copyright problems, these images are not displayed here. Hence, the results were as follows:

- 0.4 threshold:
  - The YOLO model had problems detecting the garments with darker objects, but with clearer objects, it produced a full clothing classification;
  - The Tiny YOLO model did not detect multiple clothing garments and incorrectly classified hair as a hat, and it did not detect darker objects.
- 0.2 threshold:
  - The YOLO model showed incorrect classifications or multiple classifications for a single object. However, the YOLO model classified multiple clothing garments and produce more correct classifications than the Tiny YOLO model;
  - The Tiny YOLO made multiple clothing garment classifications, but it misclassified darker objects.

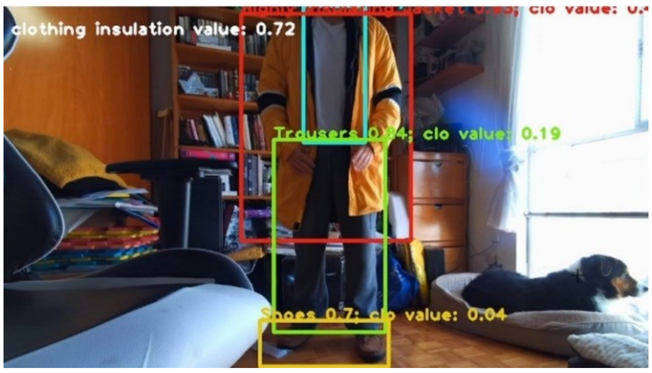
Hence, an Azure Kinect DT was employed to test the clo value in real-time. This test was oriented toward clothing insulation classification. Thus, the bounding boxes had color values depending on the clo with this assumption:

- Warmer clothing garments were closer to the red color of the bounding box; colder clothing garments were closer to the blue color.

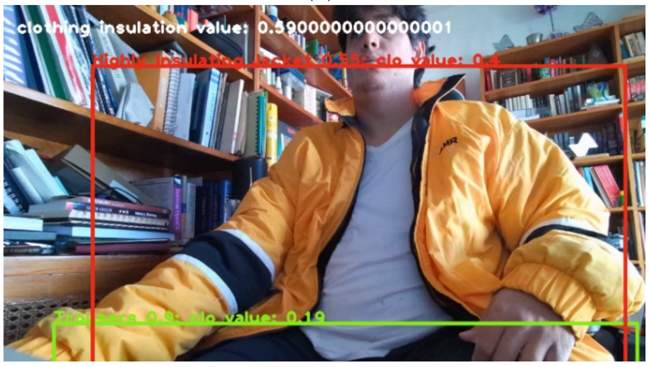
However, the Tiny YOLO model did not provide noteworthy results for multiple clothing garments recognition. Consequently, Figure 16 depicts the YOLO model results. Figure 16a shows that the model correctly classified the garments, giving a clo value of 0.32; nevertheless, it did not recognize the highly insulated jacket. Figure 16b shows that the model considered the highly insulating jacket, giving a clo value of 0.72. Figure 16c shows that the model accurately classified the highly insulated jacket and the trousers, but it did not classify the shirt. Figure 16d shows that the model correctly classified all the garments.


clothing insulation value: 0.32
Shirt 0.9; clo value: 0.09
Trousers 0.89; clo value: 0.19
Shoes 0.92; clo value: 0.04

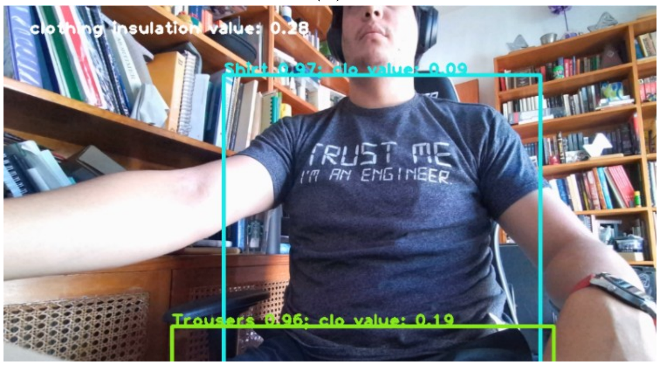
(**a**)


clothing insulation value: 0.72
Trousers 0.84; clo value: 0.19
Shoes 0.7; clo value: 0.04

(**b**)


clothing insulation value: 0.5900000000000001
Highly insulating jacket 0.55; clo value: 0.4
Trousers 0.9; clo value: 0.19

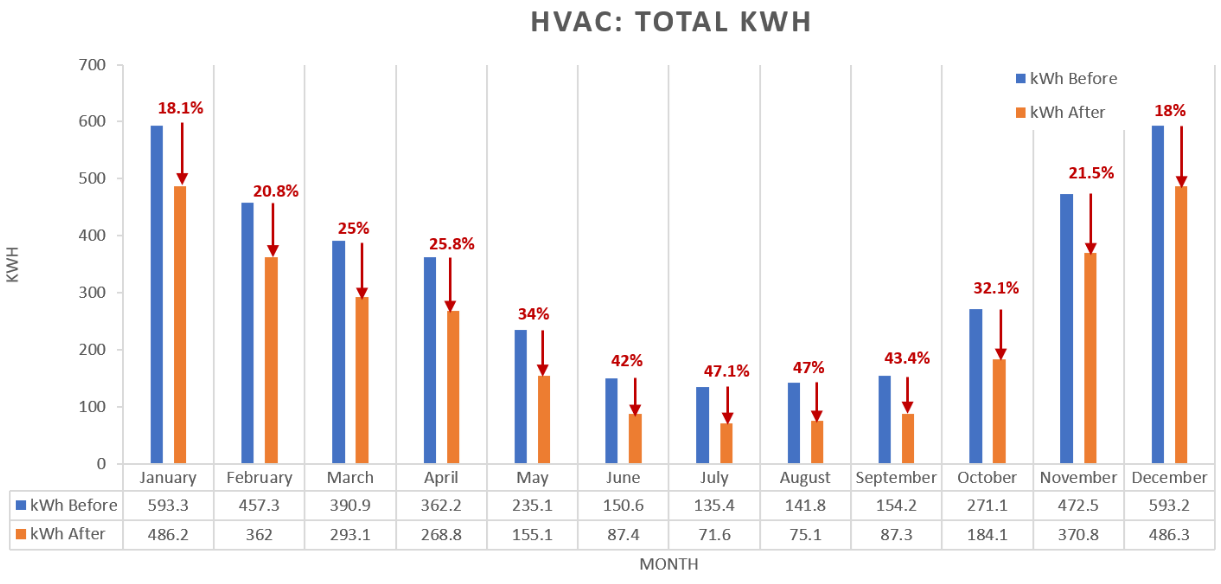
(**c**)


clothing insulation value: 0.28
Shirt 0.97; clo value: 0.09
Trousers 0.96; clo value: 0.19

(**d**)

Figure 16. YOLO classifier for clothing insulation values: (**a**) garments with 0.32 clo; (**b**) garments with 0.72 clo; (**c**) garments with 0.59 clo; (**d**) garments with 0.28 clo.

The total HVAC consumption for the living room zone was 3952 kWh. The cooling setpoint was 24.4 °C, and the heating setpoint was 21.7 °C. After increasing by 1 °C the cooling setpoint and decreasing by 1 °C the heating setpoint, the HVAC consumption was 2923.7 kWh. Figure 17 depicts the monthly chart of HVAC kWh consumption before and after increasing or decreasing the setpoint. There were monthly reductions that went from 18% to 47%. Nevertheless, strategies in the thermostat interface need to engage the householder to reduce energy consumption without losing thermal comfort.

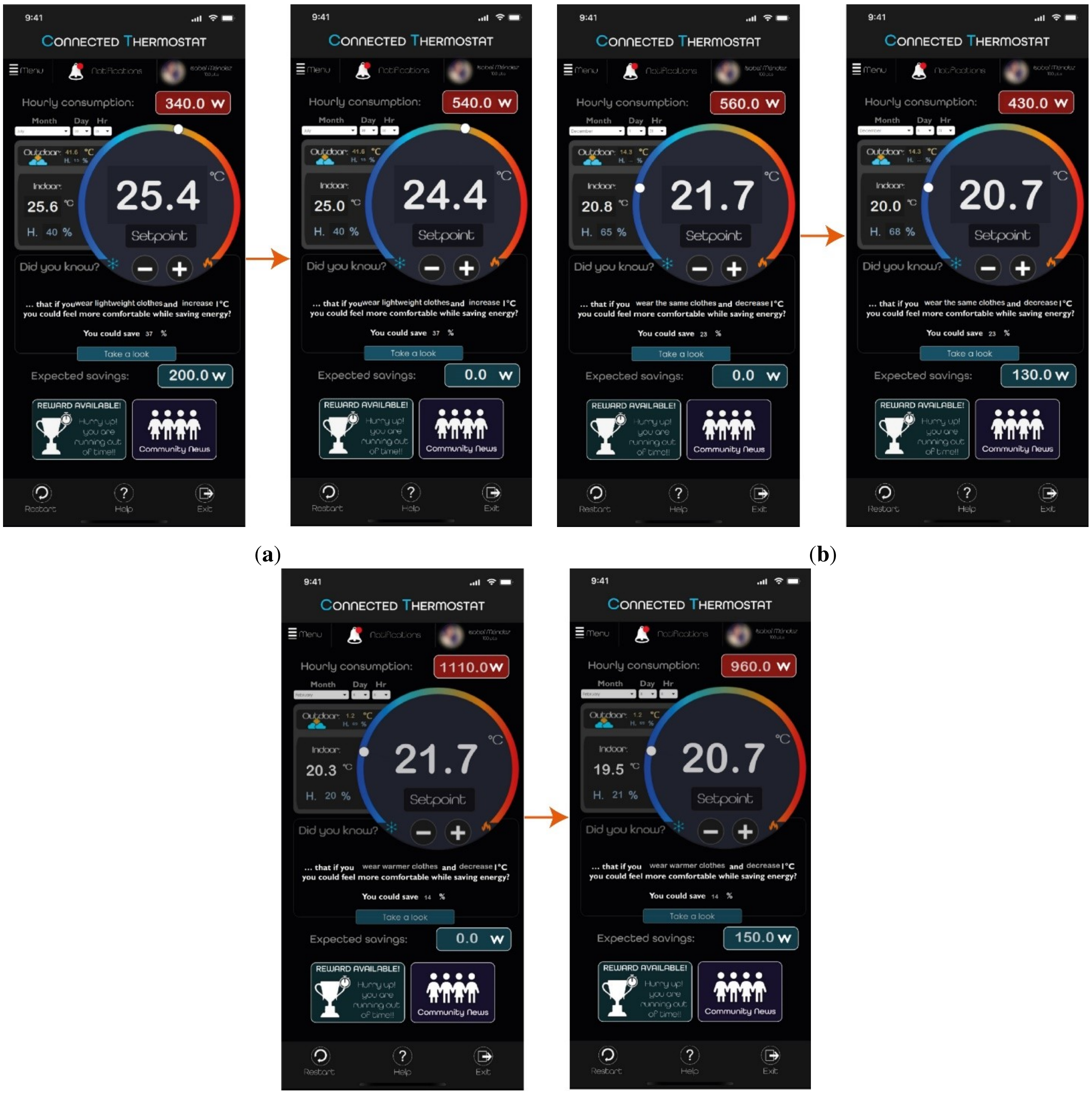


	January	February	March	April	May	June	July	August	September	October	November	December
🟦 kWh Before	593.3	457.3	390.9	362.2	235.1	150.6	135.4	141.8	154.2	271.1	472.5	593.2
🟧 kWh After	486.2	362	293.1	268.8	155.1	87.4	71.6	75.1	87.3	184.1	370.8	486.3

Figure 17. Monthly HVAC consumption in kWh.

Thus, Figure 18 displays the interface on three different dates and the required actions to reduce energy consumption:

1. July 10 at 4:00 p.m. (Figure 18a): increase the setpoint by 1 °C and wear lightweight clothes to reduce the HVAC consumption;
2. December 8 at 9:00 p.m. (Figure 18b): decrease the setpoint by 1 °C and wear the same clothes;
3. February 8 at 8:00 p.m. (Figure 18c): decrease the setpoint by 1 °C and wear warmer clothes.



... that if you wear lightweight clothes and increase 1 °C you could feel more comfortable while saving energy? You could save 37 %

... that if you wear the same clothes and decrease 1 °C you could feel more comfortable while saving energy? You could save 23 %

... that if you wear warmer clothes and decrease 1 °C you could feel more comfortable while saving energy? You could save 14 %

Figure 18. Thermostat interface actions depending on the date: (**a**) increase the thermostat temperature by 1 °C and wear lightweight clothes; (**b**) decrease the setpoint by 1 °C and stay with the same clothes: (**c**) decrease the setpoint by 1 °C and wear warmer clothes.

4. Discussion

The Fashion MNIST dataset helped as a guideline for the new dataset images. Therefore, the new dataset fitted the models' input parameters. Figure 5 shows that the printed images were correctly labeled and there was no clear bias towards a certain class after the dataset division. Accordingly, the datasets were ready to train all the CNN models.

In terms of the overall behavior, the models presented problems with the shirt, T-shirt/top, and coat classes due to the dataset containing dresses labeled as skirts. This labeling was performed to have the minimum number of classes to make the training process as efficient as possible because the increment of one class relied on 2000 more iterations for the training. Consequently, more examples are needed to avoid this confusion and improve the classifier.

The real-time implementations were successful. The real-time test for the YOLO model successfully recognized the clo values for each item of clothing and even managed to produce results in a close-up. However, at certain times, it had difficulties differentiating overlapping garments. Thus, more examples with these considerations are required. The Tiny YOLO model misclassified some garments; hence, more training images are required to make this model more robust.

Another factor to consider is that the Tiny YOLO model seemed to have no problem with computational power, but the YOLO model slowed down the real-time feed of the video. Hence, the model requires certain hardware characteristics to be successfully implemented in real-time.

Besides, real-time feedback, monitoring, and the interaction between the interface, the thermostat, and the householder allow actions to promote energy reductions without losing thermal comfort. Thus, householders can receive suggestions to increase comfort and save energy. Furthermore, to deeply understand thermal comfort and how it affects the environment and householder preferences, it is relevant to understand the type of user that is behind the interface, their preferences, and their location because their behavior will depend on other factors such as gender, age, country, culture, and fashion style, among others.

5. Conclusions

The results from the model comparison showed that the feature extraction architecture of the Tiny YOLO algorithm was on par with other image classifiers' architectures and can be used as a clothing ensemble classifier since it produced multiple clothing classifications, and it produced accuracy percentages over 90% in all three datasets, which was the objective for this project.

However, it failed to obtain better results in the independent images because the Fashion MNIST dataset had insufficient information to differentiate between the shirt class, the coat class, and the T-shirt/top class. The Tiny YOLO model only achieved 73% of correct classifications on that class, and of the remaining 27%, only 11% was misclassified as the T-shirt/top class. Thus, more information on these classes is needed since this was observed for all models, not only Tiny YOLO.

Hence, the Fashion MNIST dataset was not good enough for use as a clothing ensemble classifier since the models trained with it failed to produce more than two correct classifications in a single testing image or even obtain 90% accuracy in the independent image tests.

The YOLO model classified at least three clothing garments in real-time, but the Tiny YOLO model only produced one clothing classification 5% of the time. Hence, the YOLO model improves upon the state of the art because it outperformed the other models, giving up to four different clothing garment classifications, and consequently resulting in entire clothing ensemble recognition.

For use as a real-time clothing classifier, the YOLO algorithm is ideal as it produced results over 95% of the time in the real-time test with a threshold value of 0.5. This value was the highest obtained in the literature review for real-time implementations. Nevertheless, the Tiny YOLO model required more training examples and a greater variety of images to achieve similar results to the YOLO algorithm.

The results revealed that the new dataset proved that the model was more effective and accurate than the one trained with an existing dataset. In this new dataset, the images contained different postures to try to cover all possibilities since it hinders the accuracy of the model when the person is not in a standing posture. Furthermore, darker environment pictures need to be considered to avoid incorrect detection and classification.

Alternatively, the transfer learning method exposed that it is not ideal if the weights come from a model that is trained with a very specific dataset. Finally, the image classifier was implemented for the clothing insulation classifier during the thermal comfort calculations. Currently, the clothing garments range from 0.04 clo to 0.74 (Table 3); however, these values can be increased. Moreover, an initial assumption for underwear should be made.

The clothing insulation values are provided in the entire video; therefore, any possible changes that occur in front of the camera can be captured and considered for thermal comfort calculation, but this still leaves a gap since the system is not able to recognize the underwear the user is wearing along with any other clothing garment that the camera cannot see, making these readings inaccurate but still better than a constant value.

Hence, the clo value can be calculated in real-time, and these moving values of clothing insulation can be used in a human–machine interface, where changes in clothing garments are proposed to keep a clothing insulation value constant and allow the user to stay inside the thermal comfort ranges, but these can be equally distributed along the entire body to avoid the user feeling warm or cold due to unbalanced clothing insulation distribution.

The batch normalization eliminated the need for the dropout technique since the Tiny YOLO architecture with dropout implementation was the same as the one without it and there was no change in the accuracy scores on the training and validation datasets.

Author Contributions: Conceptualization, A.M. (Adán Medina), J.I.M. and P.P.; methodology, A.M. (Adán Medina); software, A.M. (Adán Medina); validation, A.M. (Adán Medina) and P.P.; formal analysis, A.M. (Adán Medina) and P.P.; investigation, A.M. (Adán Medina) and P.P.; resources, P.P., T.P., A.M. (Alan Meier) and A.M. (Arturo Molina); data curation, A.M. (Adán Medina) and J.I.M.; writing—original draft preparation, A.M. (Adán Medina), J.I.M. and P.P.; writing—review and editing, A.M. (Adán Medina), J.I.M. and P.P.; visualization, A.M. (Adán Medina) and J.I.M.; supervision, P.P.; project administration, P.P., T.P., A.M. (Alan Meier) and A.M. (Arturo Molina); funding acquisition, P.P., and A.M. (Arturo Molina). All authors have read and agreed to the published version of the manuscript.

Funding: This research project is supported by Tecnologico de Monterrey and CITRIS within the collaboration ITESM-CITRIS smart thermostat, deep learning, and gamification project (https:// citris-uc.org/2019-itesm-seed-funding/ (accessed on 7 November 2021)). Agreement: TECNOLÓGICO DE MONTERREY – CITRIS 2019.

Institutional Review Board Statement: "Not applicable"

Informed Consent Statement: "Not applicable"

Data Availability Statement: Data are available in a publicly accessible repository. The data presented in this study are openly available in a GitHub repository at https://github.com/zalandoresearch/fashion-mnist (accessed on 19 November 2021).

Acknowledgments: The authors would like to acknowledge the financial and the technical support of Writing Lab, TecLabs, Tecnologico de Monterrey, in the production of this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. ANSI/ASHRAE 2017 *Standard 55-2017, Thermal Environmental Conditions for Human Occupancy*; American National Standards Institute, American Society of Heating, Refrigerating and Air-Conditioning Engineers: Atlanta, GA,USA, 2017.
2. Liu, S.; Schiavon, S.; Das, H.P.; Jin, M.; Spanos, C.J. Personal Thermal Comfort Models with Wearable Sensors. *Build. Environ.* **2019**, *162*, 106281. https://doi.org/10.1016/j.buildenv.2019.106281.
3. André, M.; De Vecchi, R.; Lamberts, R. User-Centered Environmental Control: A Review of Current Findings on Personal Conditioning Systems and Personal Comfort Models. *Energy Build.* **2020**, *222*, 110011. https://doi.org/10.1016/j.enbuild.2020.110011.
4. Parsons, K.C. *Human Thermal Comfort*; CRC Press: Boca Raton, FL, USA, 2020; ISBN 978-0-429-29498-3.
5. Rupp, R.F.; Kazanci, O.B.; Toftum, J. Investigating Current Trends in Clothing Insulation Using a Global Thermal Comfort Database. *Energy Build.* **2021**, *252*, 111431.
6. Földváry Ličina, V.; Cheung, T.; Zhang, H.; de Dear, R.; Parkinson, T.; Arens, E.; Chun, C.; Schiavon, S.; Luo, M.; Brager, G.; et al. ASHRAE Global Thermal Comfort Database II. *Methods* **2021**, *2020*, 06–24.
7. Wang, L.; Kim, J.; Xiong, J.; Yin, H. Optimal Clothing Insulation in Naturally Ventilated Buildings. *Build. Environ.* **2019**, *154*, 200–210.
8. Gao, G.; Li, J.; Wen, Y. Deep Comfort: Energy-Efficient Thermal Comfort Control in Buildings Via Reinforcement Learning. *IEEE Internet Things J.* **2020**, *7*, 8472–8484. https://doi.org/10.1109/JIOT.2020.2992117.
9. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet Classification with Deep Convolutional Neural Networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105.
10. Liu, J.; Foged, I.W.; Moeslund, T.B. Automatic Estimation of Clothing Insulation Rate and Metabolic Rate for Dynamic Thermal Comfort Assessment. *Pattern Anal. Appl.* **2021**, 1–16.
11. Kalantidis, Y.; Kennedy, L.; Li, L.-J. Getting the Look: Clothing Recognition and Segmentation for Automatic Product Suggestions in Everyday Photos. In Proceedings of the 3rd ACM Conference on International Conference on Multimedia Retrieval, Dallas, TX, USA, 16–20 April 2013; pp. 105–112.
12. Yang, M.; Yu, K. Real-Time Clothing Recognition in Surveillance Videos. In Proceedings of the 2011 18th IEEE International Conference on Image Processing, Brussels, Belgium, 11–14 September 2011; pp. 2937–2940.
13. Chao, X.; Huiskes, M.J.; Gritti, T.; Ciuhu, C. A Framework for Robust Feature Selection for Real-Time Fashion Style Recommendation. In Proceedings of the 1st International Workshop on Interactive Multimedia for Consumer Electronics, Beijing, China, 23 October 2009; pp. 35–42.
14. Yamaguchi, K.; Okatani, T.; Sudo, K.; Murasaki, K.; Taniguchi, Y. Mix and Match: Joint Model for Clothing and Attribute Recognition. In Proceedings of the BMVC, Swansea, UK, 7–10 September 2015; Vol. 1, p. 4.
15. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings; Bengio, Y., LeCun, Y., Eds.; 2015

16. Liu, Z.; Luo, P.; Qiu, S.; Wang, X.; Tang, X. Deepfashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1096–1104.
17. Wang, W.; Xu, Y.; Shen, J.; Zhu, S.-C. Attentive Fashion Grammar Network for Fashion Landmark Detection and Clothing Category Classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4271–4280.
18. Zhang, Y.; Zhang, P.; Yuan, C.; Wang, Z. Texture and Shape Biased Two-Stream Networks for Clothing Classification and Attribute Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 13538–13547.
19. Lee, K.M.; Matsushita, Y.; Rehg, J.M.; Hu, Z. (Eds.) *Computer Vision—ACCV 2012*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2013; Vol. 7727, ISBN 978-3-642-37446-3.
20. Matsumoto, H.; Iwai, Y.; Ishiguro, H. Estimation of Thermal Comfort by Measuring Clo Value without Contact. In Proceedings of the MVA, Nara, Japan, 13–15 June 2011; pp. 491–494.
21. Bouskill, L.M.; Havenith, G.; Kuklane, K.; Parsons, K.C.; Withey, W.R. Relationship Between Clothing Ventilation and Thermal Insulation. *AIHA J.* **2002**, *63*, 262–268. https://doi.org/10.1080/15428110208984712.
22. Caine, K.E.; Zimmerman, C.Y.; Schall-Zimmerman, Z.; Hazlewood, W.R.; Jean Camp, L.; Connelly, K.H.; Huber, L.L.; Shankar, K. DigiSwitch: A Device to Allow Older Adults to Monitor and Direct the Collection and Transmission of Health Information Collected at Home. *J. Med. Syst.* **2011**, *35*, 1181–1195. https://doi.org/10.1007/s10916-011-9722-1.
23. Solli, H.; Hvalvik, S.; Bjørk, I.T.; Hellesø, R. Characteristics of the Relationship That Develops from Nurse-Caregiver Communication during Telecare. *J. Clin. Nurs.* **2015**, *24*, 1995–2004. https://doi.org/10.1111/jocn.12786.
24. Maharjan, R.; Bækgaard, P.; Bardram, J.E. "Hear Me out": Smart Speaker Based Conversational Agent to Monitor Symptoms in Mental Health. In *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 929–933.
25. Méndez, J.I.; Mata, O.; Ponce, P.; Meier, A.; Peffer, T.; Molina, A. Multi-Sensor System, Gamification, and Artificial Intelligence for Benefit Elderly People. In *Challenges and Trends in Multimodal Fall Detection for Healthcare*; Ponce, H., Martínez-Villaseñor, L., Brieva, J., Moya-Albor, E., Eds.; Springer International Publishing: Cham, Switzerland, 2020; Vol. 273, pp. 207–235, ISBN 978-3-030-38747-1.
26. Méndez, J.I.; Meza-Sánchez, A.V.; Ponce, P.; McDaniel, T.; Peffer, T.; Meier, A.; Molina, A. Smart Homes as Enablers for Depression Pre-Diagnosis Using PHQ-9 on HMI through Fuzzy Logic Decision System. *Sensors* **2021**, *21*, 7864. https://doi.org/10.3390/s21237864.
27. Méndez, J.I.; Ponce, P.; Medina, A.; Meier, A.; Peffer, T.; McDaniel, T.; Molina, A. Human-Machine Interfaces for Socially Connected Devices: From Smart Households to Smart Cities. In *Multimedia for Accessible Human Computer Interfaces*; McDaniel, T., Liu, X., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 253–289, ISBN 978-3-030-70715-6.
28. Avila, M.; Méndez, J.I.; Ponce, P.; Peffer, T.; Meier, A.; Molina, A. Energy Management System Based on a Gamified Application for Households. *Energies* **2021**, *14*, 3445. https://doi.org/10.3390/en14123445.
29. Méndez, J.I.; Ponce, P.; Pecina, M.; Schroeder, G.; Castellanos, S.; Peffer, T.; Meier, A.; Molina, A. A Rapid HMI Prototyping Based on Personality Traits and AI for Social Connected Thermostats. In *Proceedings of the Advances in Soft Computing*; Batyrshin, I., Gelbukh, A., Sidorov, G., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 216–227.
30. Mendez, J.I.; Ponce, P.; Medina, A.; Peffer, T.; Meier, A.; Molina, A. A Smooth and Accepted Transition to the Future of Cities Based on the Standard ISO 37120, Artificial Intelligence, and Gamification Constructors. In Proceedings of the 2021 IEEE European Technology and Engineering Management Summit (E-TEMS), Dortmund, Germany, 18–20 March 2021; pp. 65–71.
31. Mendez, J.I.; Ponce, P.; Meier, A.; Peffer, T.; Mata, O.; Molina, A. Framework for Promoting Social Interaction and Physical Activity in Elderly People Using Gamification and Fuzzy Logic Strategy. In Proceedings of the 2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Ottawa, ON, Canada, 11–14 November 2019; pp. 1–5.
32. Ponce, P.; Meier, A.; Méndez, J.I.; Peffer, T.; Molina, A.; Mata, O. Tailored Gamification and Serious Game Framework Based on Fuzzy Logic for Saving Energy in Connected Thermostats. *J. Clean. Prod.* **2020**, *262*, 121167. https://doi.org/10.1016/j.jclepro.2020.121167.
33. Méndez, J.I.; Ponce, P.; Meier, A.; Peffer, T.; Mata, O.; Molina, A. S4 Product Design Framework: A Gamification Strategy Based on Type 1 and 2 Fuzzy Logic. In *Smart Multimedia*; McDaniel, T., Berretti, S., Curcio, I.D.D., Basu, A., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2020; Vol. 12015, pp. 509–524, ISBN 978-3-030-54406-5.
34. Mendez, J.I.; Ponce, P.; Mata, O.; Meier, A.; Peffer, T.; Molina, A.; Aguilar, M. Empower Saving Energy into Smart Homes Using a Gamification Structure by Social Products. In Proceedings of the 2020 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 4–6 January 2020; pp. 1–7.
35. Mata, O.; Ponce, P.; McDaniel, T.; Méndez, J.I.; Peffer, T.; Molina, A. Smart City Concept Based on Cyber-Physical Social Systems with Hierarchical Ethical Agents Approach. In *Universal Access in Human-Computer Interaction. Access to Media, Learning and Assistive Environments*; Antona, M., Stephanidis, C., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2021; Vol. 12769, pp. 424–437, ISBN 978-3-030-78094-4.
36. Ponce, P.; Mendez, J.I.; Medina, A.; Mata, O.; Meier, A.; Peffer, T.; Molina, A. Smart Cities Using Social Cyber-Physical Systems Driven by Education. In Proceedings of the 2021 IEEE European Technology and Engineering Management Summit (E-TEMS), Dortmund, Germany,18–20 March 2021; pp. 155–160.
37. Méndez, J.I.; Ponce, P.; Miranda, O.; Pérez, C.; Cruz, A.P.; Peffer, T.; Meier, A.; McDaniel, T.; Molina, A. Designing a Consumer Framework for Social Products Within a Gamified Smart Home Context. In *Proceedings of the Universal Access in Human-Computer Interaction. Design Methods and User Experience*; Antona, M., Stephanidis, C., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 429–443.
38. Bureau of Energy Efficiency Energy Conservation in Building Space Cooling through Recommended Optimum Temperature Setting. 2018.
39. Chou, Y. *Actionable Gamification beyond Points, Badges, and Leaderboards*, Packt Publishing Ltd: Fremont, CA, USA, 2019; ISBN 978-1-83921-077-8..
40. Rawat, W.; Wang, Z. Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. *Neural Comput.* **2017**, *29*, 2352–2449.

41. Jiao, L.; Zhang, F.; Liu, F.; Yang, S.; Li, L.; Feng, Z.; Qu, R. A Survey of Deep Learning-Based Object Detection. *IEEE Access* **2019**, *7*, 128837–128868.
42. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
43. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
44. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
45. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
46. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
47. TensorFlow TensorFlow, 2021. Available online: https://www.tensorflow.org/ (accessed on 7 December 2021).
48. Huang, S.-C.; Le, T.-H. *Principles and Labs for Deep Learning*; Academic Press: Cambridge, MA, USA, 2021.
49. Team, K. Keras Documentation: Keras Layers API, 2021. Available online: https://keras.io/api/layers/ (accessed on 7 December 2021).
50. Team, K. COCO—Common Objects in Context, 2021. Available online: https://cocodataset.org/#home (accessed on 7 December 2021).
51. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv* **2017**, arXiv:1708.07747.
52. ISO. ISO 9920:2007 Ergonomics of the Thermal Environment — Estimation of Thermal Insulation and Water Vapour Resistance of a Clothing Ensemble, 2007. Available online: https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/03/92/39257.html (accessed on 2 October 2021).