Image Classification and Object Detection for Employee Monitoring in an Office Space

Bruno Castro – up202204151 & Francisco Fidalgo – up201907354

I. Abstract

In order to monitor an employee's work hours and actions throughout the day in an office space, this work sought to develop and compare several models capable of classifying behaviors such as walking, sitting, talking and sleeping, from office footage – by way of traditional machine learning and by way of a Residual Neural Network (ResNet) – and also of detecting and tracking the worker – by way of a YOLO architecture. What was seen was that, for days whose distributions were used for training the models, classification accuracies averaged 97%; for days unseen by the models, yet same office, accuracies were able to reach above 90%: ML at 92%, ResNet at 80%, and YOLO at 75%. For unseen offices altogether, accuracies were of only about 4%. After comparing different training sets, it was seen that, for the case of only one specific office, using a training set of 3000 images per class and without the inclusion of other office, the ML provided better accuracies; the same was seen when using Img2Vec features alongside HOG-based features, as opposed to only one set of features. Using YOLO, employee tracking proved to be feasible and reliable, though with lesser ACCs of those from the image classification approaches. This said, in most cases, issues of distinguishing between standing and talking (ACCs of only 10% for a given unseen day).

Overall, the traditional machine learning model, with Img2Vec+HOG features and with an SVM classifier, presented the best performance at 93% ACC for same-office unseen days, with the deep learning approaches still having great potential for improving and surpassing the former.

II. Introduction

In today's workplace environments, employee monitoring technologies have gained significant importance for several reasons. Object detection is one of the tools that have become more prevalent in recent years, especially as the rapid growth of digital technology has streamlined surveillance platform use.

One of the key factors is work security and safety since it recognizes the individuals entering and moving within the premises. With that, it is possible to prevent unauthorized access and ensure that only authorized personnel are present in specific areas. A very important aspect in the workplace on a daily basis is the need to account for employees' working hours and attendance, and a manual device/system that can read the number of hours worked by each person is often used for this purpose. With object detection, this necessity is eliminated since it provides accurate data on employee presence, working hours, and attendance patterns. Monitoring employees' productivity is another significant point to consider when managing a workplace and evaluating overall performance. To make this a simpler task, technology can be used to analyze employee movements and activities, providing insights into productivity levels and workflow efficiency. Object detection can also be useful in planning workplace occupancy, as it lets all workers know which workplaces are occupied at a given time or day. This not only improves their organization but also optimizes the workspace allocations. Employee behavior in a company is crucial and plays a significant role in organizational success. It influences many aspects, from the working environment to productivity and company culture. Object detection systems may incorporate behavioral analysis to detect anomalies or unusual patterns of activity, alerting security personnel to potential security threats or issues. Besides that, this technology can also be helpful in emergencies such as fire or security threats since it enables real-time tracking of employees, allowing for efficient evacuation and crisis management. [1]

However, workplace privacy is also a top priority. Having a tracking tool like object detection has various ethical issues that need to be addressed to ensure a fair, respectful, and lawful work environment. Employee monitoring, when done ethically and in compliance with privacy laws, helps organizations meet legal requirements and standards. [2]

A. YOLOV8 Overview

Within the computer vision domain, the refinement of object detection algorithms is pivotal for elevating the functionalities of autonomous systems. Most contemporary object detection algorithms predominantly rely on deep learning methodologies, categorized into two main groups: two-stage and one-stage approaches. A classic representation of the two-stage algorithms is the R-CNN family. This classical method initially extracts candidate frames, applies a classifier for filtration, and subsequently eliminates duplicate boxes, refining predicted boxes through non-maximal value suppression. While the two-stage detector offers advantages in terms of detection accuracy, it suffers from challenges in training complexity, slow detection speed, and optimization difficulties. On the other hand, one-stage detectors, exemplified by the You Only Look Once (YOLO) series, represent an alternative approach[3].

You Only Look Once (YOLO) has emerged as a groundbreaking paradigm shift in this endeavor. Over the recent years, numerous iterations of YOLO have been introduced. For this study, we leverage the cutting-edge YOLO V8, the

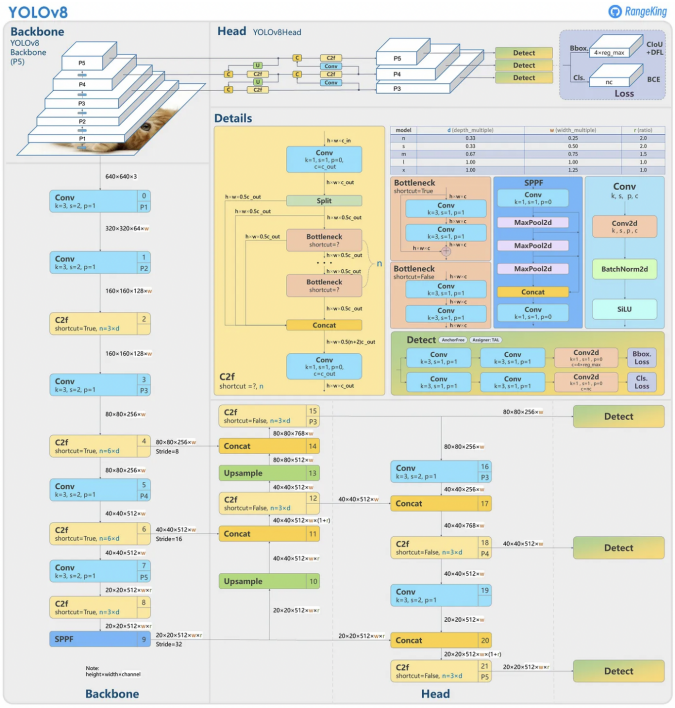

Fig. 1. Overview of the YOLOV8 architecture. Adapted from Yolov8 Architecture vs Yolov5 by E. G. Johnson

latest version in the series. YOLOv8 employs a comparable backbone to YOLOv5 but introduces modifications in the CSPLayer, now termed the C2f module. Comprising a two-convolution cross-stage partial bottleneck, the C2f module harmonizes high-level features with contextual information, elevating detection accuracy. In the output layer, YOLOv8 utilizes the sigmoid function as the activation for object scores, representing the probability of an object's presence within the bounding box. The softmax function is employed for class probabilities, indicating the likelihood of an object belonging to each possible class. YOLOv8 is selected as the baseline model for this paper, consisting of three key components: the backbone network, the neck network, and the prediction output head. The backbone network is pivotal in the YOLOv8 model, extracting features from input RGB color images. The neck network, positioned between the backbone network and the prediction output head, aggregates and processes features extracted by the backbone, crucially integrating features of different scales using a Feature Pyramid Network (FPN) structure. The prediction output head, the topmost section of YOLOv8, is responsible for identifying and locating object categories in images. With multiple detectors, each predicting position and category, YOLOv8 employs three sets of detectors with varying scales, facilitating recognition of objects of different sizes. The network architecture of YOLOv8 is depicted in Figure 1, from [4].

B. Traditional Classifiers Overview

As we delve into object detection, it becomes evident that this task inherently integrates image classification as a crucial component. Once objects are detected and localized, image classification comes into play, aiming to assign specific labels or categories to each identified object. This intersection highlights the role of several machine learning classifiers.

Support Vector Machine (SVM) is a supervised learning algorithm that aims to find a hyperplane in a feature space capable of effectively separating classes. Visualized in a two-dimensional feature space, the hyperplane represents a line whose position and orientation are determined by support vectors—points closest to the boundary between classes. The margin of the hyperplane, defined as the minimum distance between a training point and the hyperplane, plays a crucial role. SVM's primary objective is to discover the hyperplane that maximizes this margin, enhancing its generalization capability. While the algorithm assumes linear separability, it accommodates non-linear scenarios by using a kernel function, such as linear, radial (RBF), sigmoid, or polynomial, to transform the feature space [5].

K-Nearest Neighbors (KNN) is a straightforward and effective supervised learning algorithm, adaptable to binary or multi-class classification and regression tasks. It lacks fixed parameters, adjusting them based on the training data size, making it suitable for scenarios with limited prior knowledge of data distribution. Known for its "lazy" approach, KNN stores all training data and classifies new instances based on the majority class among their k nearest neighbors. The Euclidean distance is commonly used to measure the similarity between points. In essence, KNN excels in simplicity and versatility, making it a practical choice in various classification scenarios[6].

Decision Tree (DT) is a parametric model, defined across the entire input space, learning parameters by considering all training data. It operates as a hierarchical supervised learning model, constructing a tree to classify instances based on feature values. Each node represents a decision (test condition) on the instance's feature, and each branch signifies a possible value for that feature. The tree descends based on node values, passing through branches until it reaches a leaf indicating the final decision or classification category. The focus lies in determining the best feature for node classification [7].

Some algorithms are constructed from more than one decision tree are referred to as ensemble methods. One such example is the Random Forest (RF) algorithm, characterized by its random selection of features. This classifier predicts the class label of an input by majority voting on predictions made by a set of tree classifiers. Multiple decision trees are constructed, and for each, a subset of the training set with a specific number of samples is randomly chosen and created. Each tree classifier divides the feature space into a certain number of partitions, and the result for a given input is determined based on the partition in which the data resides [8].

Linear Discriminant Analysis (LDA) is a widely employed method for pattern classification, initially designed for binary classification but extendable to multiple classes. It operates under the assumption that all classes are linearly separable. In the case of two classes, LDA constructs a single hyperplane in the feature space and projects the data onto it to maximize the separation between the two categories. The creation of this hy-

perplane follows two simultaneous criteria: first, maximizing the distance between the means of the two classes, and second, minimizing the variation within each category. Essentially, LDA seeks to find the optimal linear discrimination function that effectively distinguishes between different classes by creating hyperplanes based on the underlying distribution of the data [9].

Naive Bayes classifiers adopt a "naive" approach by assuming independence between all pairs of features. The Gaussian Naive Bayes (GNB) classifier, specifically, assumes that the features follow a Gaussian distribution. Despite oversimplifying the true nature of data relationships, GNB classifiers have demonstrated surprisingly strong performance in various real-world problems. Notably, these classifiers demand only a small amount of training data being worth to highlight that the classifier's simplicity and efficiency make it a practical choice for situations where the independence assumption holds, and data exhibits Gaussian-like distribution patterns [10].

C. ResNet-18 Overview

When tasks become complex, traditional classifiers may struggle to handle them. Hence, there arises a need for a more capable classification approach. Before the introduction of the ResNet architecture, researchers faced challenges in training deep neural networks with a substantial number of layers. The main hindrance was the occurrence of the vanishing gradient problem during the backpropagation process. Existing architectures struggled to efficiently update kernel values as the number of layers surpassed a certain threshold, constraining the ability to utilize deep neural architectures with an increased number of layers. This limitation led researchers to resort to smaller network architectures. However, these smaller networks were unable to effectively discern between similar objects with intricate attributes or multiple features. Consequently, the use of such compact architectures hindered the extraction of intricate features from an image, thereby restricting the learning capabilities of neural network architectures. To tackle this issue, various ResNet architectures have been developed. In this study, we utilized one of the simpler variants, ResNet-18. The structure of the original ResNet-18 is depicted in Fig. 2.

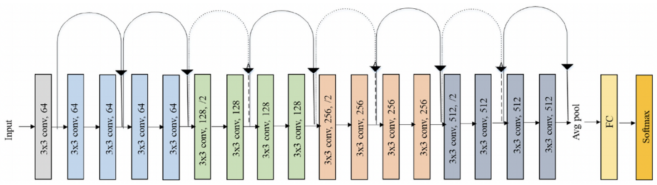

Fig. 2. Overview of the ResNet-18 architecture. Adapted from [11]

This network comprises a total of eighteen layers, including 17 convolutional layers, a fully-connected layer, and an additional softmax layer for classification. The convolutional layers employ 3×3 filters, and the design ensures that layers with the same output feature map size have an equal number of filters. However, if the output feature map is halved, the number of filters in the layers is doubled. Downsampling is achieved through convolutional layers with a stride of 2. The network concludes with average-pooling followed by a fully-connected layer incorporating a softmax layer. Throughout the network, residual shortcut connections are inserted between layers, consisting of two types. Solid lines represent connections used when input and output share the same dimensions, while dotted lines indicate connections utilized when dimensions increase. The latter type of connection performs identity mapping with zero-padding to accommodate increased dimensions, using a stride of 2 [11].

III. Methodology

A. Dataset

In order to develop a model capable of identifying if a person within the office space is present, and which actions are being taken by said person, an adequate dataset containing cases where the worker is either working or doing something else is necessary. To that end, a collection of security-camera recordings of four office spaces throughout 20 days was used for this work, created by the University of Edinburgh and available online at [12]. As described therein, 20 videos were captured at 1 frame-per-second (with a total of 456714 frames; 134110 with no one in the room; 249556 with one person in; etc.), as 1280×720 pixel color images, involving four different subjects in four different spaces, and with said subjects displaying certain behaviours (empty room; person standing/walking; person sitting; two or three people talking to each other; person has fallen – thus, five possible classes).

For convention purposes, the first office is called in this work 'Office 1', whose recordings extend from day 1 to day 12. Office 2 recorded on days 13 and 14; Office 3 on days 15, 16 and 17; and Office 4 on days 18, 19 and 20.

B. Annotations and Labels

As the supervised models being trained require labelling, with each frame of the recordings there must also be an associated label. In the case of object detection, these are instead annotations describing class and coordinates.

Bounding boxes (BB) and their labels for these behaviours are already provided by the dataset's source (in MAT format); however, for proof of concept, an initial part of this project involved the creation of some bounding boxes by the members of this group, as is further discussed in subsection IV.

As this work was developed within Python, the MAT format provided for each annotation required some pre-processing: using *scipy* for loading MAT files, each one's format was read (and turned into arrays) and changed from the original

```
(column coordinate (x) of pixel starting
at top left of bounding box around person;
row coordinate (Y) of pixel starting at top
left of bounding box around person; width of
bounding box; height of bounding box; class)
```

to the YoloV8-compatible one:

```
(class; x-coordinate of BB's center;
y-coordinate of BB's center; width of BB;
```

`height of BB)`.

To do this, it was ensured that x-coordinate of BB's center = column coordinate starting at top left of BB + width/2 (analogous process for y-coordinates); furthermore, because YoloV8's coordinate system works on a [0,1] range, normalization of the dimensions was necessary: height values divided by 720 and width by 1280, as the frames have a 1280×720 resolution.

To save these Python-compatible changes, each annotation's array was saved to a TXT file with the same name as that of the frame it corresponded to (*i.e.*, for frame called 'inspacecam163_2016_02_25_16_08_44.jpg', a TXT file was created called 'inspacecam163_2016_02_25_16_08_44.txt').

This was done so that YoloV8 could be implemented: given directories of the images folder and labels folder, each image file must have the same name as the associated text file.

To note that some TXT files had several rows, one for each "object" identified therein.

With this, some further folder rearrangements were done: as the ResNet architecture also required that each image (for training and testing) be inside a folder named after its label, for each one, the "class", found within its TXT file, was read and it determined the placement of said image into one of the five folders (five classes).

C. Data Split

In this work, only one office (Office 1, recorded from days 1 to 12) was focused on: this means that, with the intention of being able to track the worker's behavior within the office, most of the training set is from the days that recorded Office 1.

For training and validating, the data was used from the same distribution (*i.e.*, same days).

That said, as the idea is to be able to track an individual's behavior throughout a day for which there was no training data involved, days 6 and 12 (from Office 1) were excluded from being within the training and validation data distribution. Days 6 and 12 (Office 1) were chosen to better assess if training with days before and in-between may impact prediction results.

For simplification purposes, and to evaluate if one single office's dataset is sufficient for monitoring and classifying within-office behavior, most of this work uses only data from Office 1, though some exploration with data from the other offices was also done, as is addressed in section V.

Nevertheless, the data split was always **80% for training** and **20% for validation** (or same-distribution testing data in the case of traditional machine learning (TML) – since TML's model did not use validation data for training, as opposed to the deep-learning models), chosen based on the empirical assessments that an 80/20 split is best to avoid overfitting [13].

To note that, while it would have been possible to include more annotated/labeled data (found in online image databases, for example), this work only sought results from this singular dataset, for assessing the feasibility of such simplified approach.

D. Training Models

As discussed in previous sections, this work explored workspace monitoring through various approaches: Image Classification using Deep-Learning (DL IC); Image Classification using Machine Learning (DL ML); and Object Detection using Deep-Learning (DL OD): for which ResNet, ML models and YoloV8 were adopted and their configurations (*e.g.*, parameters, number of epochs) are discussed in the following subsections.

E. ResNet-based Image Classification

Leveraging essential PyTorch libraries, the implementation of the ResNet-18 model for image classification was grounded in meticulous decisions regarding both the choice of the loss function and the optimization hyperparameters. The use of the cross-entropy loss function was deliberate, leveraging its aptness for classification tasks, particularly those encompassing multiple classes. Given that ResNet-18 integrates a softmax activation function in its final layer to generate class probabilities, the cross-entropy loss seamlessly aligns with the task of gauging dissimilarity between predicted and actual distributions.

Concurrently, the stochastic gradient descent (SGD) optimizer was applied, as outlined in a comparative study by [11], with a carefully selected learning rate of 0.001 and a momentum of 0.9. The decision regarding the learning rate and momentum reflects a common starting point in many deep learning tasks, striking a nuanced balance between achieving convergence speed and ensuring stability during training.

Due to constraints on computational resources, an initial training phase of 10 epochs was undertaken, and evaluations were conducted on the 6th and 12th days to assess the model's potential. Subsequently, following satisfactory performance, a more extensive training regimen involving 100 epochs was initiated, and comprehensive testing was performed over the entire 6-day span to further refine the model's capabilities.

F. Traditional Machine Learning-based Image Classification

In exploring how robust a traditional ML model can be for classifying images, several aspects were considered: features extracted and used; ML classifier employed.

1) Feature Extraction and Selection: A first set of features extracted from the images – the called **"Img2Vec" set** – was acquired using the *img2vec_pytorch* package, obtaining vector information from each image. This "img2vec" process is one that uses a ResNet model to obtain values that may better describe the image and may enhance a ML model's ability to learn about its characteristics.

From each image resulted 511 features, all of which used for training.

A second feature set – the called **"HOG" set** – was also extracted, by using several Histogram of Oriented Gradients (HOG) features (using the *skimage.feature* package) as well as some Gabor Kernel filter features (using the *cv2* package), which can capture discriminative characteristic of an image, being often used for face recognition [14].

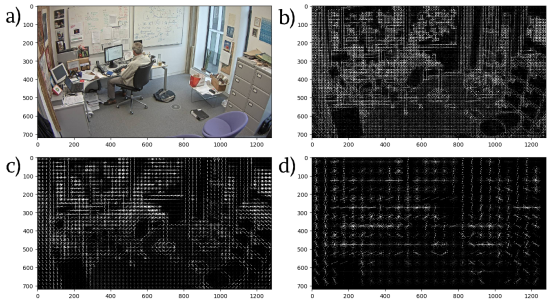

Fig. 3. HOG-transformed images of (a) a frame from the dataset, when using cell sizes (b) 8×8, (c) 20×20, and (d) 50×50.

In selecting the parameters for HOG features, it was taken into account how much useful information could be extracted depending on size. As seen in Figure 3, for a given frame (3a), as the cell size increases, the information quality decreases. In finding an adequate balance between detail and computational resources, the 20×20 cell size (3c) was chosen for HOG extraction.

From each image, over 2000 features resulting from this set were used for training.

A third set of features joined the first two – being called **"Img2Vec+HOG" set**.

To note that no elaborate feature selection process was conducted (using statistical methods such as Chi-square, ANOVA F-value, and Mutual Information scoring techniques, for instance).

2) Classifiers Used: With all features gathered (in Python, a *Pandas* DataFrame was used for arranging all features within an organized table that could then used as the training dataset; and another as the test dataset, along with the "target" (y) values (entries that correspond the classes).

Using the "scikit-learn" library in Python, the several ML models were imported — the ones considered in this work are the Decision Tree (DT), 200-estimators Random Forest (RF), linear-kernel Support Vector Machine (SVM), Linear Discriminant Analysis (LDA), K-Nearest Neighbors (KNN) and Gaussian Naïve-Bayes (GNB) classifiers. After which, each model was fitted with the training set, thereby training the model. From that, determining how adequate the model is is done by checking how often it is able to correctly classify the test set (the "accuracy").

To note that, in this work, k-fold cross validation was not performed on the data (with *cross_val_score*, which performs data splitting process on k different sections of the entire dataset).

G. YoloV8-based Object Detection

For object detection, YoloV8 was used with the default settings (learning rate of 0.001; batche size of 64; IOU threshold of 0.5; 64 convolutional filters; 53 layers; LeakyReLU activation function; AdamSGD optimizer), being considered as adequate parameters. When training, two evaluations were made: one after 30 epochs; another after 180 epochs (150 epochs on top of the first 30).

Due to lack of computational resources, exploration of hyperparameter tuning, or training for more epochs, did not occur.

H. Classification Model Evaluation

To evaluate and compare results, the accuracy score from each model is evaluated and, in this work, F1-Score, Recall and Precision were used as performance evaluation metrics.

Days 1 and 11, some frames of which were used in training, were evaluated, so as to assess the trained model's ability to classify behavior during the same days as training. Additionally, days 6 and 12, never being trained, were used as a basis for how well the model is able to classify behavior of unseen data. Days 17 and 20 were also never trained and, in cases where Offices 2, 3 and 4 were not used in training, the two were still tested to better assess how generalizable the trained models are.

IV. Proof of Concept

In the context of object detection, labels and annotations play a crucial role in training machine learning models to recognize and locate objects within images. For this purpose, the Computer Vision Annotation Tool (CVAT) online platform proved to be a very useful tool in annotating and labeling the data.

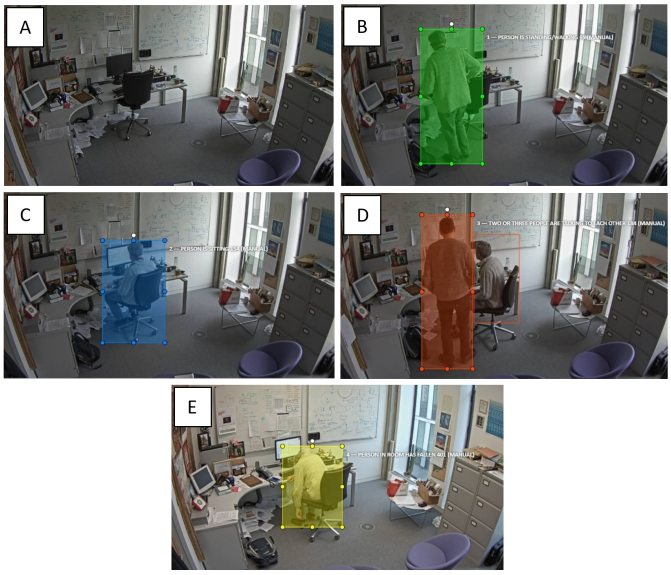

Fig. 4. Bounding boxes for the cases of (A) empty room (no BB), (B) person standing, (C) person sitting, (D) two people talking (2 BBs), (E) fallen worker [12].

For the sake of the proof of the annotation-creation concept, a random set of images from the dataset were collected, related to day 11. Considering that there were many images, only a few of them (around 100 images) were submitted to the annotation process. To accomplish the proposed task, bounding boxes were defined. A bounding box is a rectangular box that encloses an object or region of interest within an image. It is defined by its coordinates (x, y, width, height), typically represented as the top-left and bottom-right corners.

The bounding boxes should be linked to a specific label characteristic of a particular employee position/movement. Therefore, the bounding boxes should be manually drawn around the proposed objects, serving as ground truth for training object detection models – as was done using the CVAT.ai platform. It was ensured that YoloV8-compatible formatting of resulting TXT files was obtained: two specific directories were created containing the name ”images” and ”labels”, one with the images and another with the labels of same name but different extension. Once completed, this data could be used to train the model. Figure 4 shows annotated bounding boxes for examples of each of the labels being A the label 0, B the label 1, C the label 2, D the label 3 and E the label 4.

Label information is summarized in the Table I.

TABLE I
DATASET LABELS.

Labels	
0	Room is empty
1	Person is standing/walking
2	Person is sitting
3	Two or three people are talking to each other
4	Person in room has fallen

V. RESULTS

A. Traditional Machine Learning

1) Img2Vec Features: In trying to determine which dataset would be most adequate for training (how many images; from how many differing offices; etc.), several attempts were made, as summarized in TABLE II. To note that this quantity selection was made by resorting to SVM-trained classifications which, as shown later, is the classifier that seemed to provide the best overall accuracy results.

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT MODIFICATIONS ON THE TRAINING DATASET USING IMG2VEC FEATURES ACROSS MULTIPLE DAYS

Data Modifications	Classifier	Test Data	F1-Score		Recall Label 1
			Weighted Average	Accuracy	
Optimal Training Set	SVM	Validation	0.99	0.99	0.97
		Day 6	0.90	0.92	0.10
		Day 12	0.93	0.93	0.34
		Day 17	0.04	0.03	0.02
		Day 20	0.03	0.02	0.01
Multiple-People Class		Validation	0.98	0.98	0.93
		Day 6	0.90	0.92	0.10
		Day 12	0.93	0.93	0.32
Smaller Training Set		Validation	0.98	0.98	0.96
		Day 6	0.92	0.94	0.12
		Day 12	0.80	0.84	0.09
		Day 17	0.03	0.01	0.04
		Day 20	0.03	0.02	0.01
Training Set with More Offices		Validation	0.98	0.98	0.97
		Day 6	0.87	0.90	0.08
		Day 12	0.94	0.94	0.32
		Day 17	0.82	0.83	0.33
		Day 20	0.68	0.75	0.05

When using the Img2Vec set of features, a first set used only around 300 images per class (called ’Smaller Training Set’), and only of Office 1. What was seen was that, while validation results may have been adequate, reaching 98% accuracies (ACC), the same could not be said for day 12, which only ever reached 84% overall (though day 6 showed better results, despite many misclassifications of label 1, as the 12% in “Recall Label 1” shows).

Because of this, the size of the training dataset was increased to attempt to better learn about the Office 1’s characteristics. With around 3000 images per class, on the called “Optimal Training Set”, ACC results for day 12 improved significantly, by about 13%. Still, misclassification behavior was seen for label 1, often thought to be label 3 instead (Recall ACC of 10% and 34% for days 6 and 12, despite overall ACCs). This trend is the reason TABLE II, as well as several others in this work, display the Recall ACC of label 1 – to better evaluate how often this label is correctly identified.

This can also be understood from Figure 5, which shows that, while no deviations happen for validation data, nor very significantly for days 1 and 11 (days from same training distribution, but not in training data), a stark deviation does happen for day 6, wherein the model believes someone standing is, instead, someone talking. On day 12, this happens less so, though many ’standing’ misclassifications happen as well (for 427 frames, or about 7 minutes), being thought of a ’sitting’ (to note that one singular frame was annotated as ’fallen’, so, even though 100% of the ’fallen’ labels were technically misclassified, it is not enough for concluding about the model’s ability to detect label 4).

For days 17 and 20 (only the former being shown in Fig. 5), the model was very clearly not equipped for classifying behavior found within offices different from Office 1. With that, ACCs never even surpassed 4% (and even this 4% is likely a ’by-chance’ result).

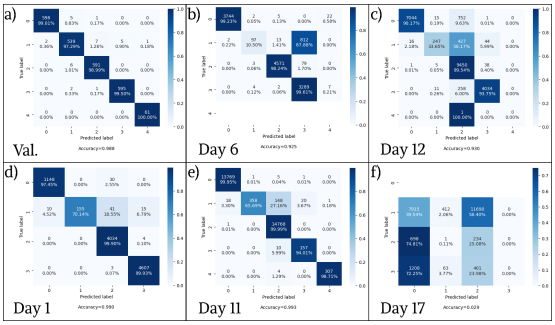

Fig. 5. Normalized confusion matrices for the Img2Vec set using the SVM classifier on data from (a) validation, (b) day 6, (c) day 12, (d) day 1, (e) day 11, and (f) day 17.

To explore whether results from days 6 and 12 (Office 1) could still be improved by supplying the model training data from other offices, a called ’Training Set with More Offices’ was created (with about 50% of data from Office 1, and the other 50% from Offices 2, 3 and 4 –equivalently, the same was done for validation/test set), leading to decreased ACCs on day 6 (by 3%) and a small increase in ACC on day 12 (by 1%). Comparing it with the optimal set, however, ACCs for days 17 and 20 now increased by 65 to 80%. This shows that, while this extension of the training set did not highly affect Office 1, it did start to generalize better to the other offices. Even still, the label 1 misclassifications persisted in days 6 and 12 (and days 17 and 20).

To try and further address the label 1 misclassification

problem, a simplification of the dataset was made: while label 3 corresponds to people talking, more often than not, those people do it standing up. To avoid confusion, whenever two or more people were in the room, that was taken as an image classified as 'talking', even if someone is standing up. This differences would teach the model that someone standing up would only be label 1 if only one person is in the room. Despite that effort, results were practically exactly the same as the 'Optimal Training Set' (in fact, not many labels ended up changing, despite the new rule).

Because the 'multiple-people class set' and the 'training set with more offices' were considered as either of negligible difference or somewhat harmful to Office 1 ACC results when compared to the 'Optimal Training Set', the latter was used for the rest of the traditional machine learning study (including for the HOG set).

To note that, in all cases, validation results were always considerably high, never going below ACCs of 98%.

Regarding weighted average, this one is given because a lot of the classes found in each day are mostly of label 0, 2 and 3, with labels 1 and 4 often times being absent. Weighted average provides a better idea about the balance between precision and recall (in the case of F1-Score) for what is otherwise an imbalance between classes.

When it came to selecting the best classifier, several results were obtained, some of which shown in TABLE III.

TABLE III
CLASSIFIER PERFORMANCE COMPARISON ON IMG2VEC FEATURES ACROSS MULTIPLE DAYS.

Classifiers	Test Data	Precision	Recall	F1-Score		Recall Label 1
		Weighted Average	Weighted Average	Weighted Average	Accuracy	
GNB	Validation	0.86	0.85	0.85	0.85	0.84
	Day 6	0.88	0.85	0.81	0.85	0.05
	Day 12	0.84	0.73	0.71	0.73	0.10
LDA	Validation	0.98	0.98	0.98	0.98	0.96
	Day 6	0.93	0.91	0.88	0.91	0.09
	Day 12	0.91	0.91	0.90	0.91	0.30
DT	Validation	0.91	0.91	0.91	0.91	0.86
	Day 1	0.93	0.93	0.91	0.93	0.23
	Day 6	0.70	0.67	0.60	0.67	0.02
	Day 11	0.95	0.95	0.94	0.95	0.30
	Day 12	0.64	0.55	0.46	0.55	0.03
KNN	Validation	0.98	0.98	0.98	0.98	0.98
	Day 1	0.99	0.99	0.99	0.99	0.76
	Day 6	0.89	0.74	0.75	0.74	0.04
	Day 11	0.99	0.99	0.99	0.99	0.81
	Day 12	0.94	0.95	0.94	0.95	0.25
	Day 17	0.98	0.02	0.04	0.02	0.02
	Day 20	0.98	0.02	0.03	0.02	0.01
RF	Validation	0.98	0.98	0.98	0.98	0.90
	Day 1	0.98	0.97	0.97	0.97	0.41
	Day 6	0.89	0.81	0.75	0.81	0.04
	Day 11	0.99	0.99	0.99	0.99	0.54
	Day 12	0.84	0.77	0.69	0.77	0.05
	Day 17	1.00	0.02	0.04	0.02	0.02
	Day 20	1.00	0.01	0.02	0.01	0.01
SVM	Validation	0.99	0.99	0.99	0.99	0.97
	Day 1	0.99	0.99	0.99	0.99	0.70
	Day 6	0.94	0.92	0.90	0.92	0.11
	Day 11	0.99	0.99	0.99	0.99	0.66
	Day 12	0.94	0.93	0.93	0.93	0.34
	Day 17	0.77	0.03	0.04	0.03	0.02
	Day 20	0.99	0.02	0.03	0.02	0.01

Overall, what was seen was that SVM outperformed every other classifier, both in validation and in days 6 and 12. Followed by LDA with ACCs about 2% lower on days 6 and 12. Even so, all classifiers displayed high ACCs for validation (all above 90%, except for the GNB). Despite GNB's overall worse performance, it can still be seen that it did better at classifying days 6 and 12 than DT, which had better validation results.

When it came do classifying images of day 17 and 20, no model was able to surpass 4% ACCs (not trained for those days); and, for days 1 and 11, most averages surpassed 95% (though with recall ACCs of label 1 sometimes quite low). Once again, the recall ACCs of label 1 on day 6, and no model (other than SVM), was able to surpass 10%. Nevertheless, it is seen that most classifications, for days 1, 6, 11 and 12 are correct.

2) HOG Features: As with the study done with Img2Vec features, TABLE IV was created for comparing ACCs.

Though ommitted, it was also seen that the SVM classifier outperformed every other; also, that days 17 and 20 had very low ACCs, barely reaching 5%.

When it came to validation, results were comparable to Img2Vec's (with SVM), as were the days of trained distributions (1 and 11) – diferences of no more than 3%; though days 6 and 12 were considerably lower (by about 15% and 4% of weighted F1-Score ACC, respectively).

Despite the new set of features, the label 1 misclassifications kept occuring; even so, overall ACCs were high and never below 80%.

TABLE IV
SVM PERFORMANCE ON HOG FEATURES ACROSS MULTIPLE DAYS.

Classifier	Test Data	Precision	Recall	F1-Score		Recall Label 1
		Weighted Average	Weighted Average	Weighted Average	Accuracy	
SVM	Validation	0.99	0.99	0.99	0.99	0.97
	Day 6	0.91	0.80	0.75	0.80	0.04
	Day 12	0.91	0.91	0.89	0.91	0.14
	Day 11	0.99	0.99	0.99	0.99	0.69
	Day 1	0.99	0.99	0.99	0.99	0.70

3) Img2Vec+HOG Features: Joining both sets of features, a final result can be seen in TABLE VI (average of its weighted F1-SCore ACC). Despite more features, results only slightly improved, with differences of 1% (higher for day 6 and 12) when compared to those had with the Img2Vec set.

B. ResNet-based Image Classification

After training the ResNet model twice (30 epochs; 100 epochs – the latter leading to losses and accuracies as seen in TABLE V), results were as summarized in Table VI. What was seen was that even with only 10 epochs, results were already able to reach 88% ACC (for days 6 and 12, on average); though, when increased by 90 more epochs, the result seems to have decreased, especially for day 12, where it went from 87% to 74%. For day 6, on the other hand, it appears to have stayed at 90% ACC, still with the label 1 deviation that was seen during Traditional Machine Learning.

Regarding the train accuracy of the model at 100 epochs, a value of 78% was obtained, even though validation was of 94%, which is unusual behavior; in both cases, loss went below 1, being lower for validation than for train.

TABLE V
COMPREHENSIVE VIEW OF LOSS AND ACCURACY IN A 100-EPOCH TRAINED RESNET MODEL

Model	Train		Validation	
ResNet (100 epochs)	Loss	Accuracy	Loss	Accuracy
	0.5738	0.7819	0.1897	0.9379

C. Yolov8-based Object Detection and Tracking

When using the YoloV8 model, trained at 30 epochs and then at 180 epochs, results obtained are also synthesized in TABLE VI. In this case, validation ACCs were higher than ResNet's – though not higher than ML; with validation being higher for 180 epochs than for 30 epochs, though not by much (only about 1%).

TABLE VI
SUMMARY TABLE ILLUSTRATING AVERAGE ACCURACIES THROUGHOUT THE STUDY; 6&12 REPRESENTING SAME-OFFICE UNTRAINED DAYS; 1&11 REPRESENTING SAME-OFFICE TRAINED DAYS; 17&20 REPRESENTING DIFFERENT-OFFICE UNTRAINED DAYS

				Average ACCs
ML (SVM)	Img2vec			0.92
	Hog			0.86
	Img2vec+Hog			0.93
DL IC	ResNet	10 Epochs	6&12	0.88
		100 Epochs	validation	0.9379
			6&12	0.805
			1&11	0.97
			17&20	0.05
DL OD	Yolov8	30 Epochs	validation	0.968
		180 Epochs	validation	0.978
			6&12	0.7515
			1&11	0.9405
			17&20	0.031

Regarding predicitions of labels during "trained" days (1 and 11), ACCs averaged 94% (slightly worse than ResNet's case, with 97%); similarly, for the non-trained days 6 and 12, 75% (5% lower than ResNet) and still considerably worse than the ML case, though it is worth noting that, in the case of this Object Detection task, the model does label 1 as 3 though unlike image classification, it does not take into account the entire picture, but only bounding boxes that surround the individual – so distinguishing between cases of many people in a room or only one person who is also standing may become more difficult. Furthermore, unlike with image classification, incorrect labels regarding 'talking' will count as multiple incorrections, as more than one BB will be estimated within the frame, as further discussed in section VI. To note that, while results may seem lowers, ACCs in this model did not consider the frames for which the model correctly guessed no one was in the room. More accurately, this model's performance has higher ACCs, since not many misclassifications happened for detecting 'empty'.

VI. DISCUSSIONS AND CONCLUSIONS

Overall, results were appreciably high, with accuracies that very commonly reached the 90% during days that the model was trained on; for brand new days (same office), while it was expected a decrease, it never really went below 75% – in this work, in fact, predicitions were above 90% for the SVM classifier (ML). Even though one might have expected better results with the DL approaches, it suffices to mention that better models often come from more extensive training. Due to lack of resources, models were not further developed with both ResNet and YOLO; even still, they showed that, with less than 200 epochs, results reached considerably high accuracies, even for non-trained days 6 and 12, and without the need of intricate feature engineering, as was done with ML.

With ML, it was seen that SVM outperformed every other option, perhaps due to its ability to effectively handle high-dimensional data (as is the case with images with numerous pixels). Perhaps if features had been less picture-focused (*i.e.*, more hand-picked and related to temporal or spatial characteristics and other aspects beyond just the image itself), a different classifier would have outperformed the SVM – one could explore this hypothesis.

When it came to finding the right size for the dataset, issues of overfitting for very specific images could have arised with a bigger dataset (only of Office 1); more images of other offices might also lead to too much underfitting where the model is incapable of discriminating relevant features. An optimal dataset with 3000 images per class was considered (of only Office 1), though eventually more offices would make this model more adaptable to different behaviors one might observe even if just for Office 1. For this work, however, the Optimal dataset was considered adequate.

On the issue of overfitting, other aspects like the number of features for ML and the number of epochs for DL were considered: while feature selection process was not conducted with ML, it was still seen that from the 511 Img2Vec set features led to results very comparable to those of the over 2000 HOG set features (and for Img2Vec+HOG set's over 2511 features). Though different characteristics, it appears that the increase in number did not necessarily lead to more inability to predict same-office images, although one can expect that many more (perhaps tens of thousands) will eventually start hindering the adaptability of the model to other days. When it came to epochs, 100-200 were chosen, which appears to not have greatly affected the Yolo model, though it did seem to affect the ResNet model (as seen by the decrease of ACCs, and also as seen by the low train accuracy of TABLE V at 100 epochs, supposedly showing that the models' bias was quite high, so not even overfitting would be expected – due to lack of resources, further research on what the issue might be was not conducted).

Overfitting may perhaps explain the label 1 misclassifications often found, especially on day 6, where all trained models believe a 'person standing/walking' is actually a 'people talking'. Looking at the images themselves within the dataset, it can be seen that day 6 has a purple chair in the middle of the room, which was not present in most other days used for training; in fact, the only days that chair was present during training was when a lady wearing purple was sitting on it and talking to the someone else. One can then conclude that all models have thus interpreted that empty purple chair of day 6 as being equivalent to the lady in purple sitting on it from one of the other days. While this is a likely explanation (that would evidence some form of overfitting), it still helps

in understanding that training a model must take many more special situations into consideration – a way to better address this would be by increasing the dataset with many different cases of background, people performing more activities: in the case of only using Office 1, this would require more days of recording (perhaps months, in order to have enough representative data on what an office day in Office 1 can be like).

In terms of real-life applicability, one thing is to note: in a practical sense, and assuming that 'sitting' corresponds to 'working', most classifications were very good ways to gauge how often the worker is working (this is because the label 2 (or 'sitting') was barely ever mislabeled nor were other labels mistaken for 'sitting' – though day 12 did suffer from cases where label 1 was 50% of the time mislabeled as 'sitting', it was never substantial, since that was no more than 500 frames (*i.e.*, 8 minutes). This tells us that the training set used is indeed accurate (for Office 1) at providing an idea of how often the person is working (estimated as 'sitting'). Day 6 with the SVM classifier, for example, would say that the worker worked ('sat') for 4571+2+13+5 frames (*i.e.*, 4591 seconds, or 1.28 hours; 98.24% of which would be actually correct (an error of 1.76% is negligible and would translate in about 1.4 minutes: if one looks at predicted labels and true labels of Figure 5, it would be concluded that the model would assume the worker (79+3)-(13+5+2)=62 seconds less than he actually did). Even still, for a real-life project, it would be desired a model able to better differentiate the remaining classes, so further development would indeed be needed.

For the sake of demonstrating what one would want to attain from workspace monitoring, a portion of day 6's recording was uploaded with bounding boxes as predicted by the trained model in this work [15]. In it, it is also shown the phenomena mentioned in the section before, also shown in Figure 6: though two subjects are talking, one of them is standing up, and the trained model took it as him standing and not talking. Aspects like these make the object detection model training a bit more nuanced – perhaps adding to the model a simplified rule that takes into account how many people/labels are being predicted would solve this issue: "if two or more labels simultaneously, then it's always label 3 ('talking')".

Fig. 6. Screencapture of Office 1 displaying YoloV8's annotations for two individuals talking, on day 6. A short video demonstrating this work's YoloV8 model performance can be found in [15].

In terms of repeatability and reproducibility, one can say that the models trained herein will be only as good at classifying and tracking someone in an office space as the office space it was trained on. Interestingly however, even the trained Yolov8 model showed an ability to identify individual's behavior in other offices, as shown in Figure 7(f) where predictions are similar to the actual box in 7(e) – this, of course, is somewhat expected. Inversely, one would not expect such capability from image classification models trained on images that entirely encompasses the exact same office space (space that surrounds the individual is considered more, as opposed to with the annotations of YoloV8 – in other words, image classification looks at entire images and classifies them; such is not the case with object detection).

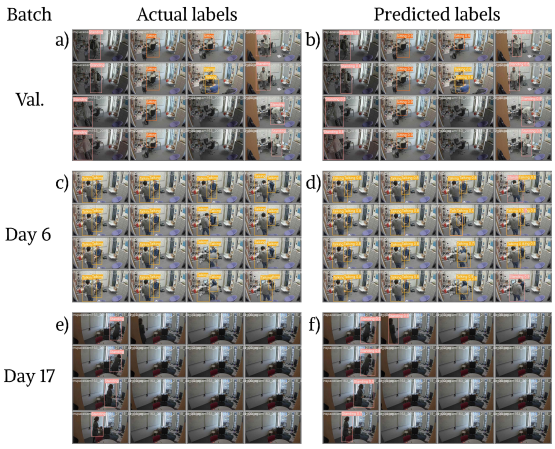

Fig. 7. YOLOV8 Classification: Ground Truth vs Predicted Labels on image batches of data from (a,d) Validation, (c, d) Day 6 and (e,f) day 17.

Even so, for better generalization, one would need to approach training the models differently: not only would there need to be more offices, but more actions of the worker would need to be captured and labeled (databases online may provide enough data samples of people talking, working, falling/sleeping and standing/walking, without even the necessity for it to be within an office space). Further should be taken to explore this expanded approach. Further work should also better optimize the DL models used: with them being the top models used today for classifying and object detecting in images, it is expected that they have the ability to provide much more robust results, provided that hyperparameter tuning is done, for instance.

With that said, it can still overall be said that, for the purpose of this work – that of being able to tell whether a person is working or not within a specific office space (Office 1) –, results were mostly adequate. With this work, it was possible to ascertain the feasibility of image classification models, both deep learning-based, by way of a ResNet, and traditional ML-based, by way of feature engineering (incorporating discriminant features from Img2Vec and HOG techniques), and also compare their performance with that of a more dedicated object identification model, by way of a YOLO architecture. What was seen was the performance by traditional ML seemed to have outperformed the alternatives; it can be expected, however, that further tinkering with the complex DL models likely leads to improved ACCs, if not even better.

REFERENCES

[1] J. Vitak and M. Zimmer, "Surveillance and the future of work: exploring employees' attitudes toward monitoring in a post-COVID workplace," *Journal of Computer-Mediated Communication*, vol. 28, no. 4, zmad007, Jun. 2023, ISSN: 1083-6101. DOI: 10.1093/jcmc/zmad007.

[2] M. Freedman. "Spying on your employees? better understand the law first." Accessed 13-01-2024. (2023), [Online]. Available: https://www.businessnewsdaily.com/6685-employee-monitoring-privacy.html.

[3] G. Wang, Y. Chen, P. An, H. Hong, J. Hu, and T. Huang, "Uav-yolov8: A small-object-detection model based on improved yolov8 for uav aerial photography scenarios," *Sensors*, vol. 23, no. 16, p. 7190, 2023.

[4] J. Terven and D. Cordova-Esparza, "A comprehensive review of yolo: From yolov1 to yolov8 and beyond," *arXiv preprint arXiv:2304.00501*, 2023.

[5] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[6] I. Saini, D. Singh, and A. Khosla, "Qrs detection using k-nearest neighbor algorithm (knn) and evaluation on standard ecg databases," *Journal of advanced research*, vol. 4, no. 4, pp. 331–344, 2013.

[7] J. Alzubi, A. Nayyar, and A. Kumar, "Machine learning from theory to algorithms: An overview," in *Journal of physics: conference series*, IOP Publishing, vol. 1142, 2018, p. 012 012.

[8] F. Noroozi, T. Sapiński, D. Kamińska, and G. Anbarjafari, "Vocal-based emotion recognition using random forests and decision tree," *International Journal of Speech Technology*, vol. 20, no. 2, pp. 239–246, 2017.

[9] J. Sarraf, P. Pattnaik, *et al.*, "Brain–computer interfaces and their applications," in *An Industrial IoT Approach for Pharmaceutical Industry Growth*, Elsevier, 2020, pp. 31–54.

[10] O. Maier, C. Schröder, N. D. Forkert, T. Martinetz, and H. Handels, "Classifiers for ischemic stroke lesion segmentation: A comparison study," *PloS one*, vol. 10, no. 12, e0145118, 2015.

[11] A. Ullah, H. Elahi, Z. Sun, A. Khatoon, and I. Ahmad, "Comparative analysis of alexnet, resnet18 and squeezenet with diverse modification and arduous implementation," *Arabian journal for science and engineering*, pp. 1–21, 2022.

[12] R. Fisher. "Edinburgh office monitoring video dataset." Accessed 20-11-2023. (2021), [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/OFFICEDATA/.

[13] A. Gholamy, V. Kreinovich, and O. Kosheleva, "Why 70/30 or 80/20 relation between training and testing sets: A pedagogical explanation," *Departmental Technical Reports (CS)*, vol. 1209, 2018.

[14] L. Shen and L. Bai, "Gabor feature based face recognition using kernel methods," in *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, IEEE. DOI: 10.1109/afgr.2004.1301526. [Online]. Available: http://dx.doi.org/10.1109/AFGR.2004.1301526.

[15] "Yolov8 test video (day 6) [video]," *YouTube*, 2023. [Online]. Available: https://youtu.be/kuh-8rg_cNw.