Visual Informatics 6 (2022) 35–50

Contents lists available at ScienceDirect

Visual Informatics

journal homepage: www.elsevier.com/locate/visinf

VisuaLizations As Intermediate Representations (VLAIR): An approach for applying deep learning-based computer vision to non-image-based data

Ai Jiang [a,*], Miguel A. Nacenta [b], Juan Ye [a]

[a] School of Computer Science, University of St Andrews, UK
[b] University of Victoria, Canada

ARTICLE INFO

Article history:
Received 31 May 2021
Received in revised form 2 May 2022
Accepted 5 May 2022
Available online 27 May 2022

Keywords:
Information visualization
Convolutional neural networks
Human activity recognition
Smart homes
Data representation
Intermediate representations
Interpretability
Machine learning
Deep learning

ABSTRACT

Deep learning algorithms increasingly support automated systems in areas such as human activity recognition and purchase recommendation. We identify a current trend in which data is transformed first into abstract visualizations and then processed by a computer vision deep learning pipeline. We call this VisuaLization As Intermediate Representation (VLAIR) and believe that it can be instrumental to support accurate recognition in a number of fields while also enhancing humans' ability to interpret deep learning models for debugging purposes or for personal use. In this paper we describe the potential advantages of this approach and explore various visualization mappings and deep learning architectures. We evaluate several VLAIR alternatives for a specific problem (human activity recognition in an apartment) and show that VLAIR attains classification accuracy above classical machine learning algorithms and several other non-image-based deep learning algorithms with several data representations.

1. Introduction

The use of machine learning and deep learning methods to classify sensor input is now commonplace and widespread over a broad set of application areas (e.g., Patel and Shah, 2019; Thomaz et al., 2012). These methods offer increasingly improved classification accuracy based on training algorithms that leverage the statistical regularities found in training data to build models that can then classify situations based on sensor states that have never been encountered by the system.

For example, we are interested in using machine learning (ML) classifiers to detect and classify human activities based on the input from sensors distributed around a home (Bianchi et al., 2019). Deploying sensors with a pretrained model in a home can support a range of desirable applications where the system uses the recognized human activity to react appropriately (a kind of implicit interaction (Cook et al., 2013)). For example, a smart home can regulate environmental factors such as temperature or lighting that are appropriate to the activity (cooking requires much light, but chatting around the table could benefit from lower, more intimate lighting). Perhaps more critically, this kind of system can assist the elderly or people with chronic conditions, disabilities or cognitive impairments by alerting of important skipped activities (e.g., taking the daily medication), detecting falls and accidents, or further helping diagnose the evolution of these conditions (e.g., Alberdi et al., 2018).

Accurately classifying activities with sufficient granularity to enable sophisticated applications remains a challenge. When binary sensors are used, this is compounded with the lack of obvious ways to integrate the location of the sensor and the timing of its activation for the classification algorithm. For example, when a user is wandering vs. working in the bedroom, the same sensors might be activated, resulting in hard-to-separate sensor features and leading to low accuracy in distinguishing these two activities (Ye et al., 2015).

Inspired by the research field of visualization (e.g., Chegini et al., 2019; Manovich, 2011; Stoiber et al., 2022), we transform the raw data into visual representations that are then used to train a vision-based deep learning algorithm. This approach, which we call *VisuaLizations As Intermediate Representation* (VLAIR), allows us to encode spatial and temporal information of sensor onsets in a straightforward and human-readable manner. We leverage knowledge in the field of information visualization to create the visualizations. The intermediate visual representation of the data is accessible to both machines and humans, because they are visualizations that may be more understandable by the human visual system than raw sensor

* Corresponding author.
E-mail address: aj99@st-andrews.ac.uk (A. Jiang).

https://doi.org/10.1016/j.visinf.2022.05.001

data. This offers humans a common representation that might be more beneficial to exploring certain classifications and supports debugging of the algorithms and deployments. We also leverage the computer vision algorithms to uncover spatial and temporal patterns exhibited in the visualizations; that is, we employ a Convolutional Neural Network—CNN, a Long Short-Term Memory network—LSTM, and an attention mechanism.

Our contribution is twofold: we introduce the VLAIR technique and we show, through a series of experiments on three datasets, how it can increase accuracy for HAR from binary sensor data. Classification improvements are of direct value to developers of human activity recognition (HAR) applications. Furthermore, because of the flexibility and the large remaining room for sophistication in the design of visualization mappings, we believe that VLAIR can be applied beyond binary sensor activity recognition and deliver further classification performance gains in this and other domains. We also provide preliminary recommendations on how to apply it based on our own experiences.

2. Background and related work

VLAIR is an alternative approach to applying machine learning (ML) and deep learning (DL) algorithms to classification problems. It leverages existing computer vision and information visualization techniques. In this section we start by relating our work to traditional approaches. Because we apply VLAIR in first instance to human activity recognition problems and, specifically, to human activity recognition through binary sensors, we review these areas separately with a special focus on their application of ML and DL techniques. We then discuss image-based applications of ML/DL that we consider precursors or examples of VLAIR and closest to our work.

2.1. Computer vision and traditional approaches to classification

Computer vision (CV) is one of the areas where DL has produced some of the most impressive recent advances, such as object classification accuracy above human performance in images (e.g., LeCun et al. (2015)). One of the most common architectures within computer vision pipelines is Convolutional Neural Network (CNNs), which were first inspired by the human visual system (LeCun et al., 2010). CNNs are particularly suitable for multi-layered machine learning because of their translation invariance and shared weights architecture, which keep the numbers of connections relatively low and hence make the classification of inputs with large number of features (or pixels) tractable (LeCun et al., 2015).

CNNs are straightforward to apply to problems where the input is directly arranged in a grid, such as the typical CV problem of recognizing objects in raster images. In problems without a straightforward spatial arrangement of data CNNs can still be surprisingly useful (e.g., Fawaz et al., 2019) but data needs to be rearranged into an appropriate shape, which might not be trivial.

When the data has a temporal component, such as in human activity recognition from sensor networks, DL architectures might include Recurrent Neural Networks (RNNs) (Graves et al., 2013). These are particularly well suited to time-based problems because they recurse through temporal signals of arbitrary length while sharing parameters across different points in time. Specifically, Long Short-Term Memory networks (LSTMs) (Hochreiter and Schmidhuber, 1997) have greatly increased accuracy for time-dependent problems such as speech recognition (e.g. Graves et al. (2013)).

2.2. Classification of human activities through ML/DL

Our work is motivated, and first tested, on applications from the area of human activity recognition. Much work in this area has been devoted to applying ML/DL algorithms to data obtained from sensors deployed in living spaces to classify activities by their occupants. This typically involves collecting and integrating data from sensors, extracting features from the raw data (Hammerla et al., 2016), and applying learning techniques to infer human behaviors. Various algorithms, including decision trees, support vector machines and, more recently, deep neural networks (Wang et al., 2019), have been applied to classification, recognition and segmentation tasks. Deep learning can demonstrably learn complex correlations between low-level sensor data and high-level human activities. For example, Morales and Roggen (2016) have employed a CNN to extract features from raw accelerometer signals and an RNN to learn sequential relationships of extracted features in human activities; Radu et al. (2018) have designed a multimodal architecture for integrating sensor data from different modalities to infer activities, and; Sprint et al. have employed change detection on Fitbit's time series data to track changes in physical activities during inpatient rehabilitation (Sprint et al., 2017).

2.3. Sensor data visualization

Some existing techniques transform raw input sensor data into spatial representations that are learnable through CNNs. For example, an early data-driven approach (Zeng et al., 2014) treats each dimension of accelerometer signals as a channel of an RGB image to capture local dependencies of sensor signals, and extracts scale-invariant sensor features with a CNN to infer human activities such as 'walking' and 'drink when standing'. Other similar approaches are to adapt 1D sensor signal inputs to form 1D virtual images and then leverage the advantages of CNNs to automatically extract and learn discriminative sensor features (Pourbabaee et al., 2017; Wang et al., 2017).

Ha et al. (2015) have combined all dimensions of sensor input forming an image and use a 2D kernel to effectively capture spatial dependency over sensors as well as local dependency over time. They take into account two different modalities: sensors in different positions and different sensing types. They group sensors in different positions to capture spatial dependency over signals via the 2D kernel and separate sensor types by padding zeros between them. Compared with using a 1D kernel, their 2D kernel method can obtain distinguishable features from multiple sensors; e.g., accelerometers, gyroscopes and magnetometers, and get better performance on common human activity recognition tasks (Jiang and Yin, 2015; Ravi et al., 2016). However, it is still challenging to overcome the need for large amounts of annotated training data due to the use of supervised deep learning techniques.

Singh et al. (2017) have addressed this limitation by utilizing the knowledge from CNNs pre-trained on image data for their sensor-based classification task. They linearly transfer 2D pressure value mappings from force-sensitive resistor fabric sensors into gray-scale images. By using a pre-trained CNN as a feature extractor, they unify the feature extraction process for pressure sensor data to better identify users from their footsteps. Our previous work also visualizes binary event sensor data in a smart home environment into color images and applies a customized CNN for the classification task (Jiang et al., 2020). We consider this work a precursor of the VLAIR approach since we use also the transformation of data into images to let a computer vision algorithm perform classification. VLAIR can be seen as a generalization of this approach in which data does not need to be strictly

spatial (as in the grid-arranged pressure map of Singh et al. (2017)), and can instead be more abstract, visually representing characteristics of the data that are not explicitly spatial in nature (e.g., the duration or sequence of activation of sensors).

2.4. Image-based transformation of abstract data for learning

Another stream of the research is to visualize traffic data into heatmaps for the prediction task. Zhang et al. (2017) have generated heatmaps to visualize inflow and outflow of grids in a region based on mobile phone signals; that is, how many people leave or enter a grid during an interval. They have applied deep spatio-temporal residual networks (ST-ResNet) for crowd flow prediction. Similarly, Zeng et al. (2020) have also generated choropleth maps on tax transaction data and applied the same ST-ResNet for traffic prediction. More importantly, this work has allowed interactive visual exploration which benefits domain experts to perform visual analytics and collaboratively develop deep traffic prediction models. Li et al. (2018) have visualized Multiplayer Online Battle Arena (MOBA) game data into a matrix and employed interactive analytics with machine learning to analyze the game performance and collaboration.

More recently, Chen et al. convert the software application's binary files into grayscale bitmap images that were then fed into a deep learning algorithm for classification as malware (Chen, 2018). They argue that, by leveraging mostly a pre-trained computer vision deep learning model that just trains a fully connected layer at its end, they can achieve higher accuracy classification than with traditional ML methods. They consider this an application of the concept of transfer learning, because the network is pre-trained with natural images but then applied to a different domain (malware detection). A few other similar examples follow a similar approach, but use different kinds of visual images. For example, Hatami et al. (2018) have transformed time-series data into recurrence plots and then apply CNNs, obtaining better performance than with an SVM classifier on SIFT, Gabor and LBP features. They have followed the work by Wang and Oates (2015), which have used Markov Transition Fields and Gramian Angular Field images to train Tiled Convolutional Neural Networks. Ahmad and Khan (2018) have processed depth data from a Kinect sensor into Sequential Front view Images and inertial data into signal images to feed AlexNet (Krizhevsky et al., 2012) and a CNN in parallel to classify human activities.

We consider the above work as examples of the VLAIR approach because they leverage transformations into images of data that is non-spatial in nature to be able to apply computer vision algorithms (see Section 5 for a more detailed explanation of the approach). However, in this paper we will argue that visualizations are not limited to this type of frequency transformations and can leverage instead experience from the area of visualization as applied for humans. Simultaneously, our work argues that there is value in sharing representations that are both interpretable by the human visual system and machine learning algorithms.

3. VLAIR definition and main terms

We define the VisuaLizations As Intermediate Representation (VLAIR) as the approach to Machine Learning that uses *abstract* spatialized transformations of data (i.e., visualization mappings) to generate bitmap images that are used as input to computer vision learning algorithms. This approach is applicable to classification problems (the main focus of this paper), but could also be applied to other tasks such as clustering (i.e., unsupervised learning). The bitmaps that are generated from the data using the visualization mappings are *visualizations* of the data and, collectively, form the *intermediate representation* of the data referred to in the name of the technique. The visualization mappings that determine how the data is transformed into visualizations can be designed by a visualization expert (a human) or automatically through an algorithm.

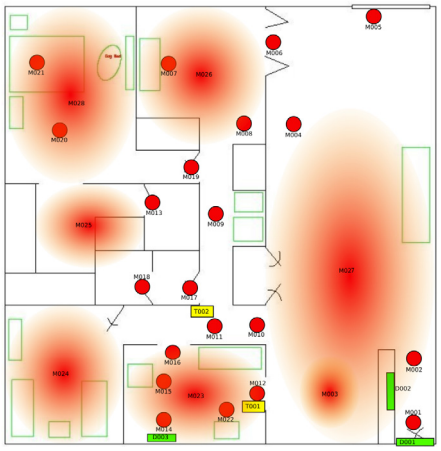
The mappings can be designed to optimize the ML task but, crucially, can also be optimized for human perception, or both. Specifics about how mappings affect visual efficiency are in the next Section. The intermediate representations are consumed by the generated CV model to perform automatic classification (or other tasks). The CV model is, in turn, the output of the training process specified by the computer vision deep learning pipeline, which uses also (possibly labeled) intermediate representations as input.

4. VLAIR visualizations

As discussed above, the core of the VLAIR technique is to transform the raw data into visualizations. This section details how we transformed the data from our specific domain into visualizations. A very large number of combinations of mappings are possible (Xiong et al., 2021), therefore a visualization is determined by a designer (human or machine) through their choice of mappings, which in turn, determines the effectiveness of the visualization for certain observer tasks. For example, designers might choose to map the dimensions of the data that they want to emphasize to the horizontal and vertical positions of visual objects (e.g., circles, dots, lines) in the plane, which have been shown to be the most powerful *visual variables* (or *channels*) for human perception of quantitative data (Heer and Bostock, 2010). VLAIR's main difference from ordinary visualization is that one of the observers is a machine-vision algorithm rather than a human; nevertheless we used simple mappings that we know would be reasonably acceptable for people as a starting point. The rationale is that since the structure of a convolutional neural network is inspired by the human visual cortex (Fukushima, 1980), visual stimuli that are known to work for humans could remain efficiently perceivable by the similarly structured CV algorithms. Understanding, optimizing and automating the choice of mappings for the differences between humans and algorithms is a promising avenue for future research, but out of the scope of this paper.

4.1. HAR and sensor data

The section will briefly introduce the problem of HAR and the type of sensors used, which motivates our visualization techniques. HAR is a classification problem in which the input is the timestamped stream of data from sensors and the output is, for any given point in time, the name of the activity that was being carried out at that time (usually from a pre-determined set of existing activities). In this paper, we mainly focus on binary event-driven sensors, which report '1' or 'ON' when being activated. Examples include RFID sensors that are activated when a tag is in close proximity (Logan et al., 2007), infra-red passive motion sensors being activated when a user is in front of them (Cook and Schmitter-Edgecombe, 2009), or switch sensors that indicate the state of physical objects, such as whether a cabinet door is open or closed (van Kasteren et al., 2011). These sensors can unobtrusively monitor users' activities and can be deployed on a wide range of objects. The raw sensor data consists of a temporally ordered sequence of binary sensor events that are annotated with activity labels. Fig. 1(a) presents an example of a deployment of these binary sensors in a home setting and Fig. 1(b) presents an excerpt of sensor data and annotated activity labels.



Date	Time	Sensor ID	Sensor Value	Activity Label
16/10/2009	03:55:53	M021	ON	Bed_to_Toilet begin
16/10/2009	03:55:58	M021	OFF	
16/10/2009	03:56:05	M028	ON	
16/10/2009	03:56:07	M028	OFF	
16/10/2009	03:56:08	M020	ON	
16/10/2009	03:56:09	M028	ON	
16/10/2009	03:56:11	M028	OFF	
16/10/2009	03:56:13	M028	ON	
16/10/2009	03:56:18	M020	OFF	
16/10/2009	03:56:20	M028	OFF	
16/10/2009	03:56:22	M025	ON	
16/10/2009	03:56:35	M025	OFF	
16/10/2009	03:56:38	M013	ON	
16/10/2009	03:56:50	M013	OFF	
16/10/2009	03:58:14	M013	ON	
16/10/2009	03:58:26	M013	OFF	
16/10/2009	03:58:28	M025	ON	Bed_to_Toilet end
16/10/2009	03:58:36	M025	OFF	
16/10/2009	03:58:42	M028	ON	
16/10/2009	03:58:44	M020	ON	Sleep begin
16/10/2009	03:58:53	M021	ON	
16/10/2009	03:58:53	M020	OFF	
16/10/2009	03:58:53	M020	ON	
16/10/2009	03:58:53	M028	OFF	
16/10/2009	03:58:54	M021	OFF	

(a) Spatial layout (b) Raw sensor data

Fig. 1. Spatial layout and the raw sensor record of the CASAS Milan dataset (Cook and Schmitter-Edgecombe, 2009).

4.2. Data pre-processing and organization

We consider two main visualization approaches. The first, and simplest, which we refer to as *static visualizations*, segments sensor events into 60-second slices and generates a single image for each slice that represents the full time slice. Previous work (e.g., Krishnan and Cook, 2014) has found that 60 s are appropriate for this kind of classification task in this kind of data; smaller periods do not capture sufficient events to successfully differentiate activities, and longer periods are detrimental to timely prediction and may contain data from multiple activities. The advantage of this approach is that the learning architecture required to process a single image is simple. Note that representing timing within the time slice is still possible. For example, it is possible to color different elements of the visualization that represent events according to how early in the 60-second period they took place.

The second approach, which we refer to as *animations* splits each 60-second slice into a further six sub-intervals which are visualized separately in the same way as with *static visualizations*, but are fed as a sequence or animation of images to the machine learning architecture. This approach is inspired by techniques of small multiples and animation used to visualize dynamic systems in visualization (e.g., Tversky et al., 2002). Animations encode timing more explicitly but they require a DL architecture that can process temporal signals (we discuss the requited architectures in detail in Section 5). Fig. 3 illustrates the difference between the two approaches.

4.3. Visualization design (mappings)

We iteratively developed a series of six initial visualization types in collaboration with one of the authors, who is a visualization practitioner and researcher. An initial session of approximately 1 h with the visualization researcher provided the base for an original set of four basic visualization types. An additional two visualization types were designed with further suggestions from fellow visualization experts.

The mappings are chosen to represent the features of the data that we find most promising a priori (e.g., the sensor layout, the activation ratios of the sensors, the sequences of activation) with visual variables that are most effective for humans according to best knowledge in information visualization (Stoiber et al., 2022) and empirical research (Liu et al., 2021). Position in the 2D plane usually ranks top in lists of visual channels ordered by efficiency and accuracy, therefore all our visualizations map the position of sensors to the location of visual elements in the 2D visualization (the 2D location visual variable). This is also a mapping that has been used in the past for a similar purpose (Singh et al., 2017) and that is understandable by human observers.

The mappings and visualization types presented below are only a small fraction of what is possible; they provide an initial informed guess of what can work, based on what works for humans. All our visualizations are based on assigning the spatial layout of sensors to horizontal and vertical position in the image. We then progressively generate other variants by adding information on sequences and sensor activation ratios, and temporal information through additional visual variables (see Table 1). Many other visualizations are possible, but their systematic exploration is outside the scope of this paper.

The subsections below describe the mappings that we have tried, except for the spatial mapping already described above, which all visualizations use. For reproducibility, the code to generate each visualization type is in the supplementary materials. Several mappings are combined in different ways to create the five visualization variants displayed in Fig. 2.

4.3.1. Sensor activation to circle radius

For each sensor i out of S total sensors in the pre-determined T interval we place a circle of radius r_i, determined by Formula (1), where k denotes the number of times that the ith sensor is continuously being recorded as active, and t_i^k is the time duration of the kth time segment, $N_r^{(t,t+T)}$ is the number of records in the time interval $(t, t+T)$, and r_{base} is the pre-defined minimum radius for a visiting sensor point, which is set as 2 in our design.

$$r_i^{(t,t+T)} = \frac{\Sigma_1^k t_i^{k(t,t+T)}}{T} * r_{\text{base}}, (1 \le k < N_r^{(t,t+T)}) \tag{1}$$

All mappings (Fig. 2, columns 1 to 5) use this.

4.3.2. Node transitions to width-variable traces

We encode sequences of events by drawing traces. Considering each activated sensor as a node, nodes activated consecutively draw a line between the positions of these nodes. The thickness of the line between nodes i and j varies according to Eq. (2), where w_{base} is the pre-defined minimum width (set as 2 in our design) for a line indicating one-time visit and $N_{i,j}$ is the total number of visits between sensor i and sensor j in a T-length interval.

$$w_{i,j} = N_{i,j} * w_{\text{base}} \tag{2}$$

The visualizations of columns 1 to 4 in Fig. 2 use this.

Table 1
The different mappings (rows) that are applied to the different visualization types that we have tested (columns). T stands for traces, R for Room shape shadows, DMT for Day and Minute Time, and BC for Bézier Curve.

		Visualization type				
		T	DMT	RT	RT+DMT	RT+DMT+BC
Mapping	Sensor x,y → Image x,y	✓	✓	✓	✓	✓
	Sensor Activation → Circle Radius (4.3.1)	✓	✓	✓	✓	✓
	Node Transitions → Width-variable Traces (4.3.2)	✓	✓	✓	✓	
	Node Transitions → Curvature-variable Traces (4.3.3)					✓
	Time of Day → Color (4.3.5)	✓	✓	✓	✓	✓
	Room of Sensor → Room Shape Shading (4.3.4)		✓	✓	✓	✓
	Full Timestamp → Color (R,G,B) (4.3.6)		✓		✓	✓

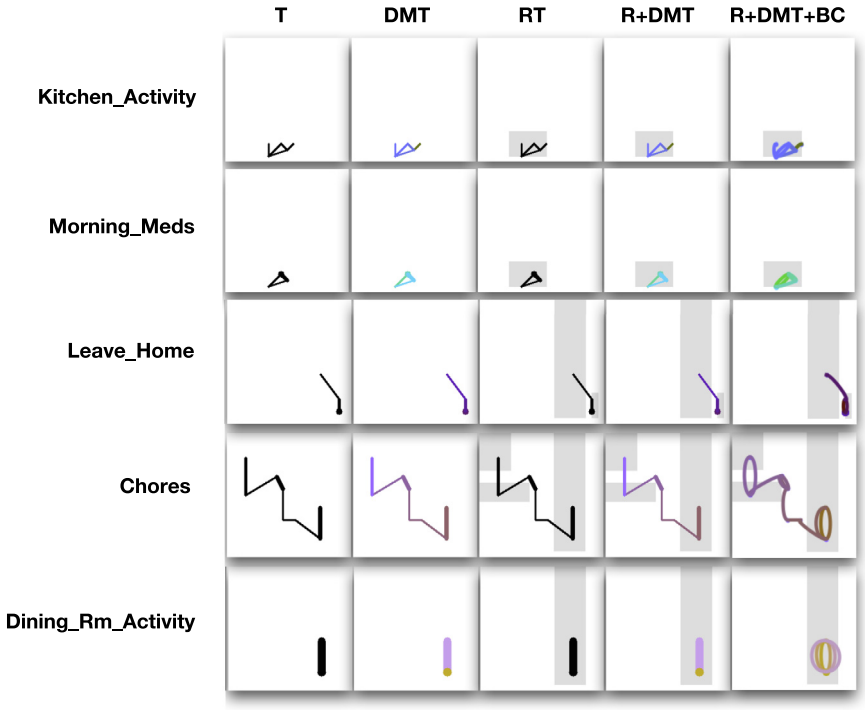


Fig. 2. Summary of the transformed images for a collection of activities.

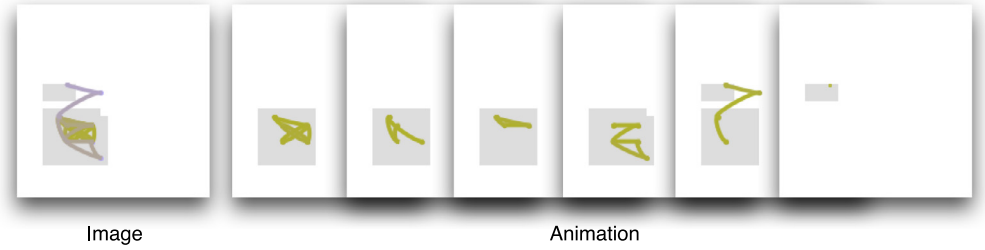


Fig. 3. Static visualization (left) compared to the sequence of visualizations with the animation approach (right) for an activity "Housekeeping" with the R+DMT+BC visualization type.

4.3.3. Node transitions to curvature-variable traces

The straight lines described in Section 4.3.2 can often overlap if there exist multiple visits between the same pair of sensors. To separate these traces, we provide an alternative that uses curved connections between sensor locations with curvatures progressively increasing when the same pair of sensors is activated multiple times. Here we employ *Cubic Bezier Curves* which is defined by 2 target points (s_1 and s_2) and their corresponding control points (c_1 and c_2). The line between s_1 and c_1 is the tangent of the curve on s_1, whose distance determines how long the curve moves into direction c_1 before turning towards s_2. Here we use the distance to encode the count of visits between two sensor points. Let $N^{[t,t']}_{s_1 \to s_2}$ be the count of visits from a sensor point s_1 to s_2 during the interval $[t, t']$, where $t \leq t' \leq t + T$. The distance between c_1 and s_1 linearly increases with this count of visits, which is defined as follows:

$$d^{[t,t']}_{s_1 \to s_2} = N^{[t,t']}_{s_1 \to s_2} * d_{\text{base}}, \tag{3}$$

where d_{base} is the pre-defined minimum distance for the control points indicating one-time visit (set as 0.1 in our design). With the increase in the count of visits, we can separate the visits between the same pair of sensor points.

With cubic Bezier curves we can encode the change rate of a sensor activation. For example in Fig. 4, the angle a_1 is between

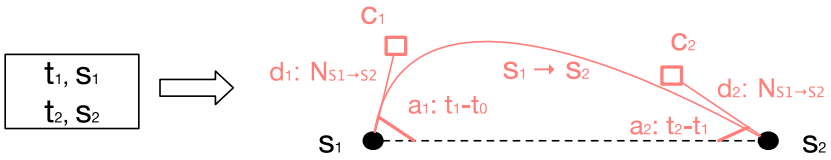


Fig. 4. A Cubic Bézier curve. The text box represents the sensor activation records: at the timestamp t_1, a sensor s_1 is activated; and the next timestamp t_2, s_2 is activated. The righthand side visualizes the corresponding cubic Bézier curve.

the line $s_1 \rightarrow c_1$ and the angle is linearly proportional to the elapsed time before the current sensor activation s_1. Let t_0 be the timestamp on a previous sensor activation, t_1 be the timestamp that a sensor s_1 becomes active and t_2 be the timestamp that another sensor s_2 is active (i.e., a trajectory from s_1 to s_2), then the angles a_1 and a_2 are calculated as

$$a_1 = \frac{t_1 - t_0}{T} \tag{4}$$

$$a_2 = \frac{t_2 - t_1}{T}. \tag{5}$$

To further make the trajectory more visible, the visits from s_2 to s_1 will go under the line $s_1 \rightarrow s_2$; that is, the same angles but minus π.

The visualization of column 5 in Fig. 2 uses this.

4.3.4. Sensor location context to room shape

Sensors are located in rooms, which are useful delimiters of human activity. We add gray shadings of the areas of all rooms which have at least one of its sensors activated in the corresponding period. RT, R+DMT, and R+DMT+BC, shown in Fig. 2, columns 3 to 5, use this.

4.3.5. Time-of-day to color

Time of day might be relevant for distinguishing activities with similar sensor activation; e.g., preparing breakfast and dinner typically occur at different times, even though they might trigger a common set of sensors in the kitchen. Therefore we encode time-of-day information by drawing all elements in the image with a color that corresponds to the time of the day. The colormap is taken from the *Matplotlib* built-in colormaps,[1] where we select 24 different color levels from blue (early morning) to red (late night). DMT, R+DMT, and R+DMT+BC, shown in Fig. 2, columns 2, 4 and 5, use this.

4.3.6. Full timestamps to color

More granular time information might be useful for distinguishing some activities from others. For example, the traces of entering and leaving the house might look similar, but the order in which sensors are activated will be reversed. In this mapping we reserve the Red channel and Green channel of the image to represent the hour of the day (24 levels) and the minute within the hour (60 levels) respectively. The blue channel represents the second within the visualization's represented period, with specific nodes being more or less blue depending on whether they are activated at the end or beginning of the period. This color coding on traces is shown in Fig. 2, columns 2, 4, and 5.

5. Machine learning architectures

Our proposed approach takes raw binary sensor data as input, segments them into fixed intervals, and transforms each segment into a VLAIR static image (static) or, alternatively, forms a short sequence of shorter static images (animation). These two sources of input have then to be processed with a machine learning workflow that outputs a model to classify static images or animations accurately. In this section we describe the main architectures and networks that we implemented. We considered three main elements to combine in the architecture: a CNN, a RNN and an attention mechanism.

Convolutional Neural Network We aimed first at a relatively lightweight architecture that could run on a resource-constrained device. A small-sized model requires relatively short training time to converge and, in our case, we only need to deal with simple images that largely consist of primitive shapes such as curves, rectangles, and circles. Driven by this purpose, we designed a CNN composed of three 2D convolutional layers each followed by a rectified linear unit (ReLU) and a max pooling layer, then a dense layer followed by a softmax classification layer.

Recurrent Neural Network Our area of application (HAR) has a temporal component, since activity patterns temporally evolve governed by the motion of inhabitants. A natural solution to be able to classify the animation visualizations is to combine static image features with Recurrent Neural Networks (RNN) for sequence pattern learning between consecutive frames (Wu et al., 2017).

We explicitly model the VLAIR animation as an ordered sequence of frames by employing a Long Short-Term Memory (LSTM) network (Fan et al., 2021) whose input is the frame-level CNN features; that is, CNN outputs are processed forward through time and upwards through stacked LSTMs. A final softmax classifier is added to make a prediction.

Attention Mechanism One important property of human perception is that humans tend not to process the whole scene in its entirety at once. As we can see in Fig. 2, the activity traces in our visualizations only take up a small area of a VLAIR image and we wanted to explore whether focusing only on the trace area would improve the activity recognition. To test that, we consider an attention mechanism that concentrates on particular areas of an image rather than treating the whole image equally (Xu et al., 2015).

A general way to do so is to use weighted image features; that is, multiplying an attention map with the image feature map. The attention map represents a positive weight for each spatial area of an image, indicating the importance of the area to the task. This is often referred to as *soft attention*. In contrast, *hard attention* only samples one area of an image to attend to at a time, rather than inputting the whole image. However, hard attention is not differentiable (Luong et al., 2015) and is computationally expensive. Soft attention allows regular and easier back-propagation, as the gradients can be directly computed through the stochastic process (Xu et al., 2015).

A more recent attention mechanism, *self-attention*, uses local receptive fields of convolutional operations (Wang et al., 2018) to avoid excessively deep networks and improve performance. This could help compute the response at a position in a sequence by attending to all positions and taking their weighted average in an embedding space.

[1] The matplotlib build-in colormaps can be access at: https://matplotlib.org/tutorials/colors/colormaps.html.

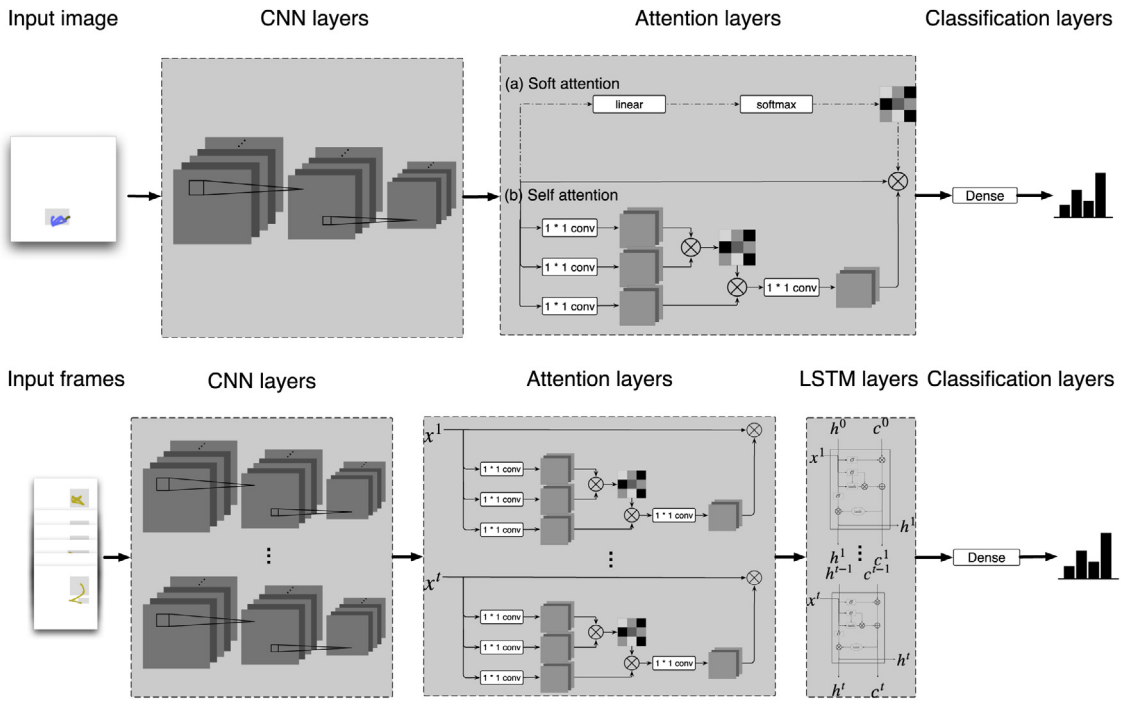


Fig. 5. Overview of CNN-based (upper) and LSTM-based (down) model architectures.

In our experiments we consider a basic soft attention and a self-attention module as a complement to the underlying CNN and LSTM neural networks as discussed above.

5.1. Overall architecture

One of the goals of our experiments is to determine what kind of DL architecture would work best for the two types of input considered (single images and animations). We combine the three architectural elements discussed above according to the type of input. For single-image classification we apply first the CNN and, optionally, attention layers (no need for temporal networks since this type of input does not contain sequences). Attention is either soft attention or self attention. A final fully connected classifier provides the activity prediction (see Fig. 5 top).

For the animation input, we connect the CNN layers, followed (optionally) by the attention layers (soft attention or self attention), followed (optionally) by the LSTM layers and a fully connected classifier at the end as well (see Fig. 5 bottom).

6. Evaluation methodology

To validate the VLAIR approach we design a series of experiments that test the different visualizations types of Fig. 2 against a collection of state-of-the-art alternatives not based on images, as well as with different supporting architectures. The experiments are designed to answer the following questions:

- **Q1** Which type of VLAIR visualization leads to more accurate recognition?
- **Q2** Which DL architecture results in best VLAIR accuracy?
- **Q3** Does VLAIR outperform existing raw sensor data-based activity recognition approaches?

The following subsections describe the methodological design choices for the experiments.

6.1. Datasets

To demonstrate our approach, we work with three HAR datasets based on binary event-driven sensors. The state-of-the-art third-party datasets are from the CASAS project, published by Cook and Schmitter-Edgecombe (2009) and represent data collected from three deployments in homes in Aruba, Milan, and Tokyo.[2] All datasets are well annotated with the same procedure (although with different categories) and contain varied user activities. The activity labels were produced by the multiple annotators of the CASAS team using the house plan, sensor positions and forms completed by the residents with information of the times and locations of their activities (Aminikhanghahi et al., 2018). Fig. 1(a) presents the spatial layout of one of the smart home settings (Milan), with the location of the sensors (marked in red circles). Fig. 1(b) lists a small subset of the raw sensor data along with their activity annotation. The activity classes and their distribution for each dataset can be found in Table 4. We additionally extracted the relative 2D coordinates of the sensor locations from the apartment layouts, which are also provided in the supplementary materials.

In the Aruba dataset, a single elderly woman lived in the apartment during 2010–2011 and performed daily activities such as meal preparation, eating, and working. She lived with a dog, and children and grandchildren visited her regularly. The Aruba testbed had 31 wireless motion sensors, 4 door sensors, and 4 temperature sensors. We only keep motion sensors and door sensors that are relevant to this study. The Milan dataset was collected in the home of a female adult and a dog volunteer in 2009 through 28 wireless motion sensors. Her children visited on several occasions. In the Tokyo dataset, the apartment housed two residents (*R1* and *R2*), who performed their daily activities including working, preparing meals, and sleeping (for this reason, we refer to this dataset as the *Twor* dataset). Real-world environments usually contain multiple users and recognizing multi-user concurrent activities is essential for scenarios such as smart

[2] The three CASAS datasets can be accessed at http://casas.wsu.edu/datasets/.

Table 2
CNN configuration.

Type	Configurations
Input	$N \times M \times 3$ image
Convolution	Filter: 64, Kernel size: 3 × 3, Stride: 1
Max pooling	Kernel size: 2 × 2, Stride: 2
Convolution	Filter: 64, Kernel size: 3 × 3, Stride: 1
Max pooling	Kernel size: 2 × 2, Stride: 2
Convolution	Filter: 128, Kernel size: 3 × 3, Stride: 1
Max pooling	Kernel size: 2 × 2, Stride: 2
Fully connected	512 neurons
Softmax	C neurons

homes. However, recognizing the activity of two people through identity-agnostic sensors is challenging, and it mainly relies on learning the subtle differences between users when they perform the same activity (Ye et al., 2015).

We deliberately select these datasets because of the density of sensor deployment and because they are well annotated with a wide range of activities. The evaluation on these datasets will help us to understand (1) whether our proposed method generalizes to different smart home datasets, and (2) how the performance compares to state-of-the-art methods when noise is present in the normal sensor readings; e.g., the dog's movement and family visits can trigger abnormal sensor activation that is inconsistent with usual activity patterns.

6.2. Comparisons

To address the questions in the previous section we have to assess how VLAIR approaches compared to non-VLAIR (baseline) approaches. Within VLAIR approaches, we also want to know whether single images work better than animations and whether attention mechanisms (soft-attention or self-attention) help. Finally, we intend to assess which of the VLAIR visualizations proposed (T, DMT, RT, RT+DMT, RT+DMT+BC) works best. To provide a fair assurance that VLAIR outperforms non-VLAIR approaches, we have to cover a sufficient range of non-image-based (non-VLAIR) approaches and architectures. Even though there exist no state-of-the-art models for direct comparison to VLAIR, we consider deep learning architectures of comparable sophistication to the ones used with VLAIR (LSTM and CNN), as well as classical supervised learning algorithms (K-Nearest Neighbors, Support Vector Machines and Random Forests). For the same fairness and generalizability reasons, we consider three types of data representations (Raw, Mutual Information, and Location+Time) for each one of the non-VLAIR algorithms (further details on the choice of baselines for comparison are in Section 6.4).

Because the number of combinations is too large to present or compare, we do not provide a fully crossed measurement of all the possibilities. Instead, we often select the best performing visualization-model-transformation-architecture of a certain category, e.g., the best VLAIR approach, and report it when appropriate.

6.3. Hyperparameter setting

Achieving good classification with image- or non-image-based DL approaches requires some fine tuning of the parameters and hyperparameters of the algorithms and architectures. We follow the state-of-the-art fine-tuning methodologies on all the approaches, and more details can be found in our supplementary file. We summarize our setting in Table 2.

Table 3
Key statistics of the datasets and their corresponding baseline representations.

Dataset	Feature Dimension			No. of Instances	No. of Classes
	RAW	MI	LOC+TIME		
Aruba	31	31	913	217,407	11
Twor	43	43	847	82,283	23
Milan	28	28	414	69,372	15

6.4. Baselines

Regarding non-VLAIR approaches, we consider baselines from two orthogonal dimensions of variation: data representation and model type. Data representation refers to the feature set that is provided to the machine learning algorithms. We consider three alternatives: raw (RAW), location and time (LOC+TIME), and Mutual Information (MI). The RAW representation contains activation intensity for each sensor in each interval, as described in Eq. (6).

$$p_i = \begin{cases} \frac{N_i}{\sum_{j=1}^{S} N_j} & \text{if } i \in [1, S] \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where N_i is the number of the times that the ith sensor is activated during the interval.

The LOC+TIME representation provides additional spatial and temporal information in an equivalent way to the VLAIR approaches by adding sensor coordinates, room coordinates, hour information, and transitions (equivalent to traces) information to the data already in RAW. Finally, the MI representation encodes the contribution of each sensor event based on temporal and sensor mutual information, as described in Krishnan and Cook (2014). In this approach, *temporal dependency* measures the contribution of a sensor event in a segment based on its temporal distance to the last event in the segment and *sensor dependency* measures the probability of two sensors occurring consecutively. Differently from the RAW representation, which counts sensor events, the MI feature vector weighs the influence of sensor events based both on their temporal dependency and sensor mutual information. Table 3 summarizes the key dimensions of the three representations.

The other orthogonal dimension concerns the model type, which is coupled to the classification algorithm. We consider the following model types, some of which have been previously tested on this kind of data:

- *classic supervised learning*: Naive Bayes, K Nearest Neighbors (KNN), Classification And Regression Tree (CART), Support Vector Machine with linear and RBF kernels (SVM, SVM-RBF), and Random Forests (RF). All implementations come from the scikit-learn library (Pedregosa et al., 2011). Because CART and linear-kernel SVM results are much poorer than all other approaches, we omit them from the result reporting and the discussion.
- *deep learning*: CNN and LSTM. The designed CNN is composed of three groups of convolutional layers each of which consists of a 2D convolutional layer followed by a max pooling layer, a dense layer with 512 neurons followed by a dropout layer, and a softmax classification layer. Besides CNN, we also design a LSTM which is placed after the first fully connected layer of CNN. We have experimented with various numbers of layers and memory cells, and chose to use three stacked LSTM layers, each with 512 neurons. We use the same preprocessing criterion for the raw sensor data. That is, n individual sensor segments are inputs into n convolutional networks which are then connected to a three-layer LSTM, in which the output from one LSTM layer

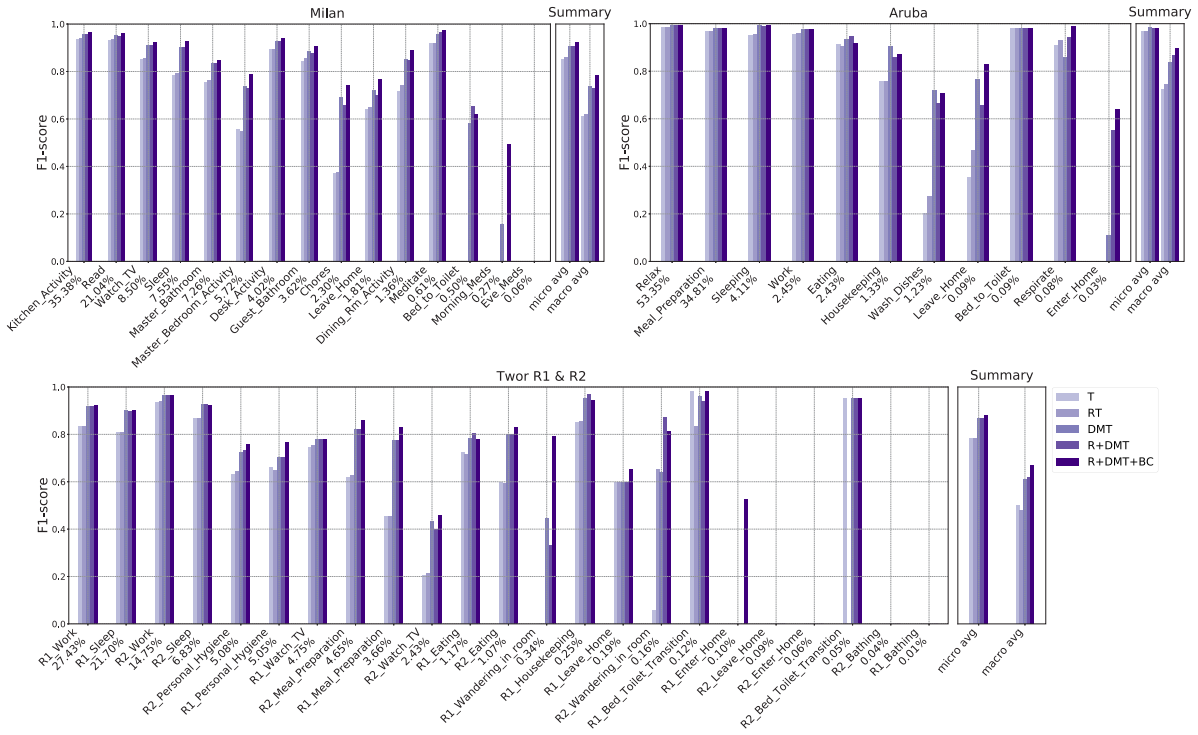


Fig. 6. Comparison of performance on different VLAIR mappings. All the displayed mappings are trained with the CNN+LSTM architecture.

is input for the next layer. A dropout layer is added and a dense layer provides the final predictions of the model.
For both CNN and LSTM, we have used the same methodology as VLAIR for grid search on the hyperparameters such as the number of neurons and layers, and learning rate. The final configuration is selected to balance the accuracy and computation efficiency.

In summary, we consider 3 classic machine learning techniques KNN, SVM, and RF, and 2 deep learning techniques CNN and LSTM. Each technique will be applied to 3 feature representations: RAW, MI, and LOC+TIME. In total, we compare VLAIR with 15 models on raw sensor data.

6.5. Validation methodology

For each of the VLAIR techniques, we run 5-fold cross validation, which is considered appropriate for long-term datasets and has been applied on the same datasets (Feuz and Cook, 2017; Ye et al., 2015). The validation set is obtained by splitting the training data (i.e., the K-1 folds) into 80% for model training and 20% for validation.

6.6. Metrics

We use F1-scores as our main accuracy measure because they balance precision and recall. More specifically, we use *macro F1-scores* (averaging the F1-scores from all activity classes) and *micro F1-scores* (averaging across all instances). For every condition we calculate the scores from the average of 5-fold cross validations. We run all experiments on the same dedicated machine: an Intel workstation with a processor i5-8500 CPU @ 3.00 GHz, 6 cores and 64 GB memory with a NVIDIA Quadro p6000 GPU, and measure execution times.

7. Results

Here we present the main findings from the experiments grouped in three subsections corresponding to the three questions in Section 6. First we address the issue of what visualization will produce the most accurate results, then which architectures support this best, and then we compare the best VLAIR results with non-VLAIR approaches. A final subsection describes our execution time measurements.

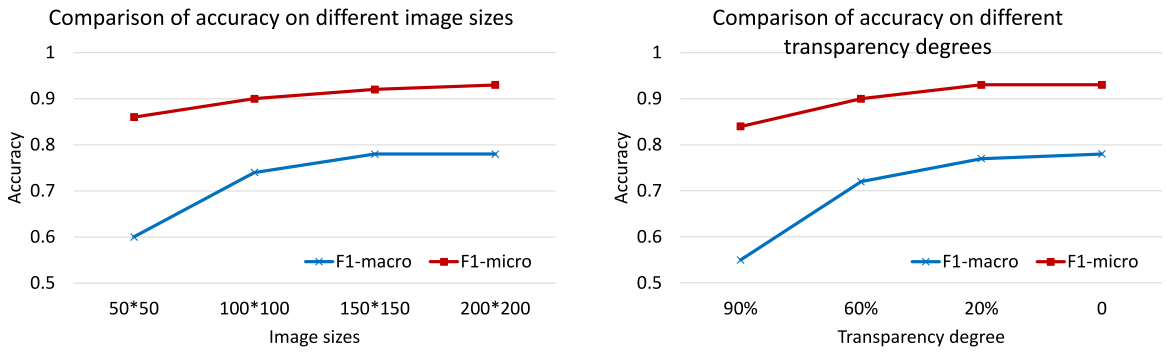
7.1. Comparisons of VLAIR visualizations (Q1)

Fig. 6 visually summarizes the F1-scores of the CNN + LSTM VLAIR architecture trained with different VLAIR mappings of the animation type.[3] We present the F1-scores on each class and the averaged micro and macro F1-scores at the end for each dataset. Among all VLAIR variants, the RT+DMT+BC visualizations offer the best micro and macro measures of accuracy for all the datasets. R+DMT+BC is most accurate in 39 of the 49 classes across all datasets.

Gray-scale sensor traces, *T* and RT, perform worst, and among them, adding room shape does not improve the accuracy much, which does not support our original assumption. There are two significant improvements in DMT (which encodes temporal knowledge of sensor traces) and in BC (which separates overlapping traces). Both of them provide more information about human movements in relation to the activity, including at what time of a day an activity is performed, between which areas the movements transition, and how often transitions between certain areas take place.

To further explore the impact of different visualization elements, we have experimented with different resolution sizes and different color transparency degrees. Fig. 7 compares F1-micro

[3] This represents the most sophisticated architecture. We selected this one for the VLAIR visualization comparisons based on preliminary tests on subsets of the available datasets.



(a) Image resolution (b) Transparency degree

Fig. 7. Comparison of performance on different image sizes and color transparency degrees.

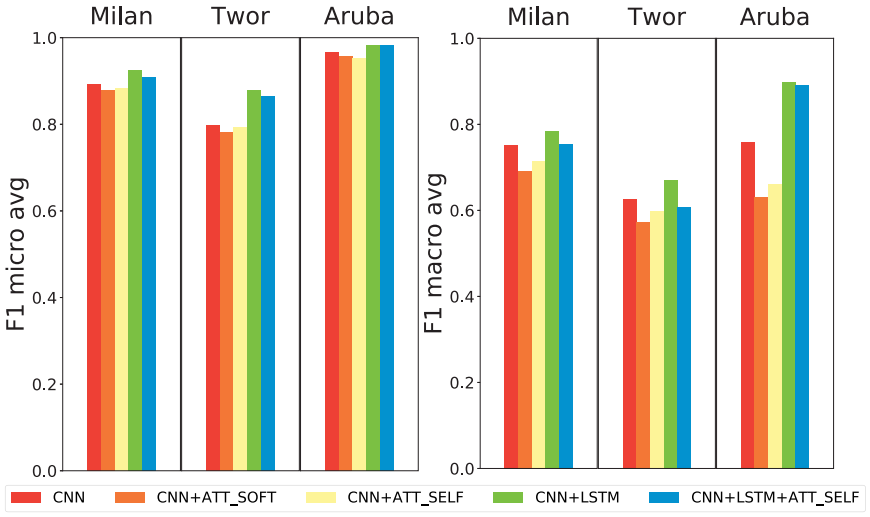


Fig. 8. Ablation analysis on each machine learning component: CNN, attention, and RNN.

and F1-macro scores with these settings on the Milan dataset. As we can see in Fig. 7(a), with the increase of image size, both F1 scores improve and converge around 200 × 200. Also the training with larger images seems more stable and converges better. We also experiment different transparency degrees on the lines of visualizations of the Milan dataset. Fig. 7(b) shows that the less transparent the lines, the higher the recognition accuracy. These two experiments have confirmed that more information in visualization will lead to better classification performance.

7.2. Comparisons of VLAIR DL architectures (Q2)

To select the most appropriate VLAIR architecture we compare the performance of the best performing visualization (RT+DMT+BC) in multiple combinations of the architectural components described in Section 5. The accuracy results from an ablation analysis are displayed in Fig. 8. These measurements correspond to architectures where certain components are removed so that we can assess the contribution of that component towards accuracy. The results show that with the RT+DMT+BC visualization the addition of the LSTM component results in increased accuracy. F1-macro scores with CNN+LSTM are 3, 4 and 13 percentual points higher than the best non-LSTM architecture on the Milan, Twor, and Aruba datasets, respectively. Against our expectations, adding an attention mechanism (self or soft) resulted in slightly reduced accuracy across the board.

7.3. Comparisons of VLAIR with baseline approaches (Q3)

We compare a selection of the best VLAIR models with the baselines in accuracy and execution times of the different computational elements of the workflow. The VLAIR architecture with CNN+LSTM on animation input has shown to be the best in the earlier experiments.

Table 4 shows a performance overview of the conditions in our experiments. We present F1-scores on each activity with activities ordered by frequency. The table shows 5 VLAIR visualizations, and 15 non-VLAIR conditions (3 data representations × 5 model types) in columns. We compare the results on each class on all the datasets and highlight the best F1-score on each activity class. As we can see, VLAIR has outperformed the other comparison techniques on most of the individual activities and the overall accuracy.

Fig. 9 summarizes this data in visual form but shows only the best variants in each category. The best accuracy overall corresponds to VLAIR. The best visualization (R+DMT+BC) has very high accuracy, slightly above than the second visualization VLAIR R+DMT, and much better than RF LOC+TIME, which was the best of the non-VLAIR alternatives by a substantial margin. The VLAIR CNN-only architecture (with single image input) is also consistently better than any of the non-VLAIR alternatives, including those with LSTM elements, but the difference is not that

Table 4
F1-scores of VLAIR mappings and baseline approaches. For each activity, we highlight the best F1-scores.

		Milan																	Aruba											
ML	FEATURE	Kitchen_Activity (35.38)	Read (21.04)	Watch_TV (8.5)	Sleep (7.55)	Master_BR (7.26)	Master_BR_A (5.72)	Desk_Activity (4.02)	Guest_Bathroom (3.62)	Chores (2.30)	Leave_Home (1.81)	Dining_Rm (1.36)	Meditate (0.61)	Bed_To_Toilet (0.50)	Morning_Meds (0.27)	Eve_Meds (0.06)	F1-Micro	F1-Macro	Relax (63.15)	Meal_Preparation (23.18)	Sleeping (6.09)	Work (2.9)	Eating (2.65)	Housekeeping (0.91)	Wash_Dishes (0.85)	Bed_To_Toilet (0.1)	Respirate (0.08)	Leave_Home (0.07)	Enter_Home (0.02)	F1-Micro	F1-Macro
VLAIR	RT+DMT+BC	**0.97**	**0.96**	**0.92**	**0.93**	**0.85**	0.79	**0.94**	**0.91**	**0.74**	0.77	**0.89**	**0.98**	0.62	**0.49**	0.00	**0.93**	**0.78**	**0.99**	**0.98**	**0.99**	**0.98**	0.92	0.87	0.71	**0.83**	**0.98**	**0.99**	**0.64**	**0.98**	**0.90**
	RT+DMT	0.96	0.95	0.91	0.84	0.73	**0.90**	0.93	0.88	0.66	**0.85**	0.70	0.97	**0.65**	0.00	0.00	0.91	0.73	0.99	0.98	0.99	0.98	**0.95**	0.86	0.67	0.66	0.98	0.94	0.55	0.98	0.87
	DMT	0.96	0.95	0.91	0.83	0.74	0.90	0.93	0.88	0.69	0.85	0.72	0.96	0.58	0.16	0.00	0.91	0.74	0.99	0.98	0.99	0.98	0.94	**0.90**	**0.72**	0.77	0.98	0.86	0.11	0.98	0.84
	R+T	0.94	0.93	0.86	0.76	0.55	0.79	0.89	0.86	0.38	0.74	0.65	0.92	0.00	0.00	0.00	0.86	0.62	0.99	0.97	0.96	0.96	0.91	0.76	0.28	0.47	0.98	0.93	0.00	0.97	0.75
	T	0.94	0.93	0.85	0.76	0.56	0.79	0.89	0.84	0.37	0.72	0.64	0.92	0.00	0.00	0.00	0.85	0.61	0.99	0.97	0.95	0.96	0.91	0.76	0.21	0.36	0.98	0.91	0.00	0.97	0.73
LSTM	LOC+TIME	0.90	0.93	0.86	0.86	0.83	0.46	0.89	0.83	0.00	0.58	0.63	0.81	0.17	0.00	0.00	0.84	0.58	0.98	0.95	0.99	0.94	0.86	0.41	0.00	0.00	0.89	0.00	0.00	0.96	0.55
	MI	0.85	0.53	0.25	0.38	0.09	0.33	0.37	0.16	0.13	0.09	0.12	0.21	0.00	0.00	0.00	0.59	0.23	0.76	0.61	0.33	0.36	0.00	0.04	0.00	0.00	0.00	0.00	0.00	0.68	0.19
	RAW	0.90	0.93	0.84	0.76	0.82	0.35	0.90	0.81	0.00	0.60	0.62	0.84	0.00	0.00	0.00	0.83	0.56	0.99	0.95	0.97	0.95	0.87	0.50	0.00	0.00	0.44	0.59	0.00	0.96	0.57
CNN	LOC+TIME	0.88	0.92	0.82	0.76	0.82	0.28	0.88	0.80	0.07	0.50	0.58	0.65	0.00	0.00	0.00	0.82	0.53	0.98	0.95	0.97	0.94	0.87	0.32	0.00	0.00	0.00	0.27	0.00	0.96	0.48
	MI	0.84	0.49	0.00	0.41	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.56	0.12	0.76	0.60	0.33	0.33	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.67	0.18
	RAW	0.89	0.91	0.84	0.74	0.81	0.29	0.89	0.80	0.00	0.49	0.56	0.59	0.00	0.00	0.00	0.82	0.52	0.98	0.95	0.96	0.94	0.87	0.30	0.00	0.00	0.24	0.40	0.00	0.96	0.51
RF	LOC+TIME	0.88	0.93	0.88	0.89	0.80	0.44	0.88	0.83	0.14	0.60	0.58	0.85	0.46	0.08	0.00	0.85	0.62	0.99	0.94	0.99	0.96	0.88	0.50	0.06	0.04	0.93	0.75	0.00	0.97	0.64
	MI	0.82	0.74	0.57	0.53	0.67	0.24	0.77	0.62	0.07	0.32	0.43	0.75	0.07	0.00	0.00	0.67	0.44	0.68	0.21	0.10	0.08	0.00	0.02	0.01	0.00	0.11	0.01	0.00	0.54	0.11
	RAW	0.88	0.93	0.87	0.83	0.80	0.35	0.89	0.82	0.12	0.62	0.52	0.81	0.00	0.00	0.00	0.84	0.56	0.99	0.95	0.97	0.96	0.89	0.48	0.01	0.04	0.86	0.73	0.00	0.97	0.62
SVM	LOC+TIME	0.88	0.94	0.87	0.82	0.80	0.29	0.86	0.80	0.00	0.63	0.56	0.79	0.00	0.00	0.00	0.84	0.55	0.99	0.95	0.97	0.96	0.88	0.45	0.00	0.00	0.78	0.00	0.00	0.97	0.54
	MI	0.46	0.66	0.55	0.31	0.54	0.00	0.72	0.36	0.00	0.00	0.24	0.67	0.00	0.00	0.00	0.49	0.30	0.68	0.22	0.10	0.11	0.00	0.01	0.00	0.00	0.17	0.00	0.00	0.54	0.12
	RAW	0.87	0.93	0.87	0.77	0.81	0.12	0.90	0.83	0.00	0.61	0.50	0.80	0.00	0.00	0.00	0.83	0.53	0.98	0.95	0.96	0.95	0.89	0.20	0.00	0.00	0.59	0.00	0.00	0.96	0.50
KNN	LOC+TIME	0.86	0.93	0.79	0.86	0.77	0.45	0.88	0.82	0.11	0.60	0.57	0.80	0.46	0.00	0.00	0.83	0.59	0.99	0.95	0.99	0.96	0.89	0.49	0.00	0.00	0.88	0.74	0.00	0.97	0.63
	MI	0.83	0.72	0.49	0.53	0.68	0.22	0.79	0.66	0.03	0.36	0.46	0.79	0.00	0.00	0.00	0.64	0.44	0.09	0.56	0.10	0.11	0.00	0.02	0.00	0.00	0.16	0.00	0.00	0.41	0.09
	RAW	0.89	0.93	0.88	0.82	0.76	0.33	0.90	0.82	0.10	0.62	0.59	0.83	0.00	0.00	0.00	0.84	0.57	0.99	0.94	0.97	0.96	0.80	0.44	0.00	0.00	0.86	0.79	0.00	0.96	0.61

		Twor																												
ML	FEATURE	R1_Work (27.43)	R1_Sleep (21.70)	R2_Work (14.75)	R2_Sleep (6.83)	R2_Per_Hyg (5.08)	R1_Per_Hyg (5.05)	R1_Watch_TV (4.75)	R2_Meal_Prep (4.65)	R1_Meal_Prep (3.66)	R2_Watch_TV (2.43)	R1_Eating (1.17)	R2_Eating (1.07)	R1_Wander_In_Rm (0.34)	R1_Housekeeping (0.25)	R1_Leave_Home (0.19)	R2_Wander_In_Rm (0.16)	R1_Bed_to_Toilet (0.12)	R1_Enter_Home (0.10)	R2_Leave_Home (0.09)	R2_Enter_Home (0.06)	R2_Bed_to_Toilet (0.05)	R2_Bathing (0.04)	R1_Bathing (0.01)	F1-Micro	F1-Macro
VLAIR	RT+DMT+BC	**0.92**	**0.90**	0.96	0.92	**0.76**	**0.77**	**0.78**	**0.86**	**0.83**	0.46	0.78	**0.83**	**0.79**	0.94	0.65	0.81	**0.98**	0.52	0.00	0.00	**0.95**	0.00	0.00	**0.88**	**0.67**
	RT+DMT	0.92	0.90	**0.97**	**0.93**	0.73	0.70	0.78	0.82	0.78	0.40	**0.80**	0.80	0.33	**0.97**	0.60	**0.87**	0.94	0.00	0.00	0.00	0.95	0.00	0.00	0.87	0.62
	DMT	0.92	0.90	0.96	0.92	0.73	0.70	0.78	0.82	0.77	0.43	0.78	0.80	0.44	0.95	0.60	0.64	0.96	0.00	0.00	0.00	0.95	0.00	0.00	0.87	0.61
	R+T	0.83	0.81	0.94	0.87	0.64	0.65	0.75	0.63	0.46	0.21	0.72	0.59	0.00	0.85	0.60	0.65	0.83	0.00	0.00	0.00	0.00	0.00	0.00	0.78	0.48
	T	0.84	0.81	0.94	0.87	0.63	0.66	0.75	0.62	0.45	0.21	0.72	0.60	0.00	0.85	0.60	0.06	0.98	0.00	0.00	0.00	0.95	0.00	0.00	0.78	0.50
LSTM	LOC+TIME	0.79	0.81	0.90	0.90	0.69	0.63	0.77	0.73	0.71	0.31	0.49	0.52	0.00	0.00	**0.72**	0.00	0.86	**0.67**	0.00	0.00	0.50	0.00	0.00	0.78	0.48
	MI	0.73	0.79	0.87	0.81	0.42	0.65	0.69	0.62	0.49	0.27	0.58	0.34	0.00	0.00	0.57	0.00	0.84	0.00	0.00	0.00	0.00	0.00	0.00	0.73	0.38
	RAW	0.79	0.80	0.91	0.90	0.67	0.46	0.74	0.71	0.00	0.00	0.57	0.07	0.00	0.00	0.56	0.00	0.74	0.00	0.00	0.00	0.00	0.00	0.00	0.75	0.34
CNN	LOC+TIME	0.78	0.80	0.90	0.90	0.63	0.65	0.76	0.69	0.65	0.34	0.60	0.57	0.00	0.50	0.70	0.00	0.86	0.54	0.12	0.00	0.80	0.00	0.00	0.77	0.51
	MI	0.72	0.79	0.87	0.80	0.48	0.64	0.69	0.63	0.31	0.03	0.54	0.29	0.00	0.00	0.00	0.00	0.43	0.00	0.00	0.00	0.00	0.00	0.00	0.72	0.31
	RAW	0.77	0.81	0.90	0.90	0.67	0.23	0.69	0.66	0.00	0.00	0.34	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.73	0.27
RF	LOC+TIME	0.76	0.82	0.90	0.92	0.60	0.66	0.77	0.68	0.63	0.57	0.59	0.54	0.07	0.39	0.51	0.00	0.93	0.25	**0.25**	0.13	0.82	0.29	0.00	0.78	0.53
	MI	0.72	0.60	0.79	0.68	0.39	0.63	0.66	0.49	0.56	0.20	0.35	0.28	0.02	0.29	0.50	0.06	0.86	0.39	0.22	**0.18**	0.52	0.00	0.00	0.63	0.41
	RAW	0.76	0.83	0.91	0.93	0.50	0.68	0.73	0.57	0.54	0.32	0.54	0.46	0.00	0.40	0.49	0.06	0.94	0.28	0.06	0.00	0.83	0.00	0.00	0.77	0.47
SVM	LOC+TIME	0.75	0.80	0.90	0.92	0.56	0.63	0.73	0.70	0.00	0.00	0.51	0.43	0.00	0.00	0.56	0.00	0.89	0.00	0.00	0.00	0.00	0.00	0.00	0.75	0.36
	MI	0.32	0.46	0.60	0.27	0.00	0.53	0.63	0.00	0.35	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.42	0.14
	RAW	0.75	0.83	0.91	0.93	0.51	0.66	0.73	0.71	0.00	0.00	0.42	0.47	0.00	0.00	0.57	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.76	0.33
KNN	LOC+TIME	0.77	0.83	0.91	0.93	0.58	0.66	0.77	0.65	0.58	**0.54**	0.55	0.50	0.00	0.34	0.47	0.00	0.95	0.30	0.08	0.15	0.90	**0.37**	0.00	0.78	0.51
	MI	0.71	0.66	0.67	0.58	0.43	0.53	0.41	0.50	0.53	0.23	0.31	0.30	0.00	0.14	0.42	0.00	0.69	0.31	0.00	0.06	0.26	0.00	0.00	0.59	0.34
	RAW	0.76	0.83	0.90	0.92	0.54	0.65	0.71	0.61	0.46	0.41	0.58	0.45	0.00	0.34	0.44	0.00	0.94	0.19	0.23	0.06	0.86	0.10	0.00	0.76	0.48

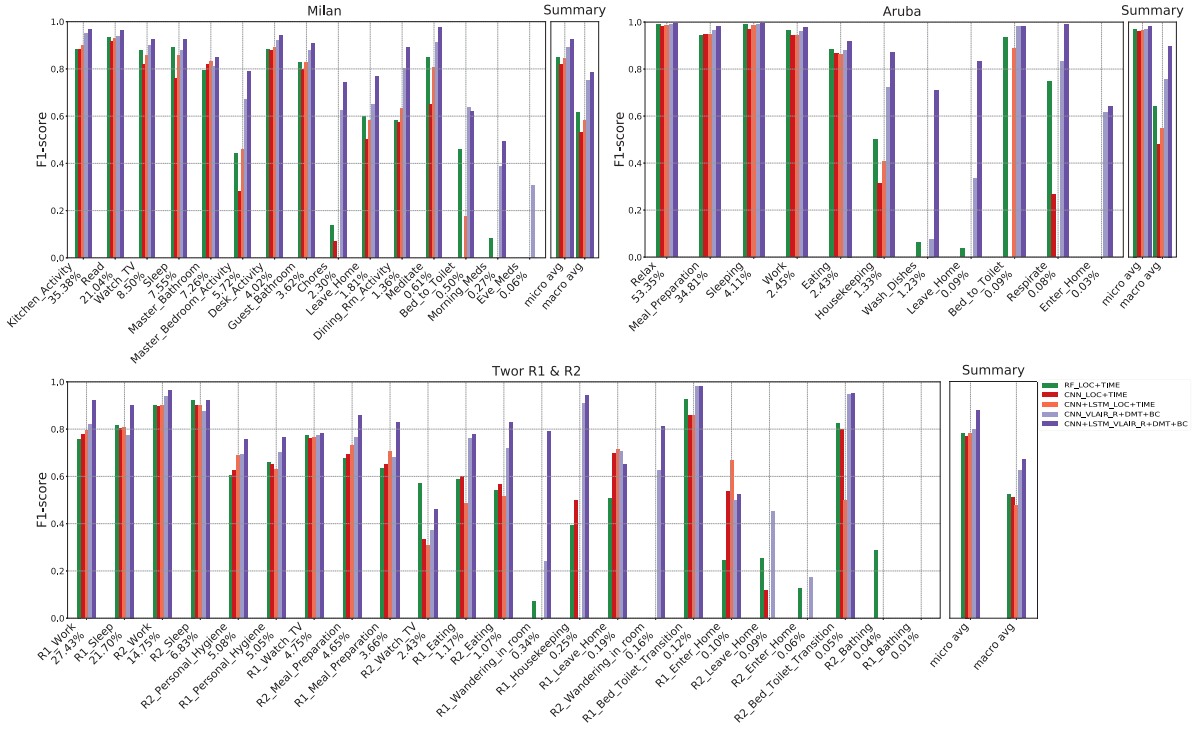


Fig. 9. F1-scores for VLAIR R+DMT+BC (violet), CNN+LSTM (orange), CNN (red) and RF (green) on LOC+TIME sensor features. Activities are ordered from left to right in order of frequency (indicated below activity name). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 5
Welch's t-test statistics and p-values comparing VLAIR and the baselines in F1-scores. ∗ means statistically significant. All tests are run in R version 3.3.2.

Baseline		Aruba		Milan		Twor	
		t(11)	p-value	t(15)	p-value	t(23)	p-value
LSTM	LOC+TIME	3.13	0.01066*	3.6	0.00288*	3.17	0.004425*
	MI	9.84	0.0000018*	9.3	0.0000002*	4.57	0.0001511*
	RAW	3.49	0.005868*	3.72	0.002279*	4.51	0.0001755*
CNN	LOC+TIME	3.63	0.004617*	4.34	0.0006727*	3.23	0.003823*
	MI	9.95	0.0000017*	8.72	0.0000005*	5.42	0.000019*
	RAW	3.75	0.003793*	4.3	0.000738*	5.2	0.0000324*
KNN	LOC+TIME	2.85	0.01737*	4.12	0.001038*	2.7	0.0131*
	MI	16.95	0*	6.62	0.0000115*	6.33	0.0000022*
	RAW	3.03	0.01271*	3.8	0.001933*	3.53	0.001872*
SVM	LOC+TIME	3.21	0.009359*	3.79	0.001987	4.38	0.0002404*
	MI	14.27	0.0000001*	8.24	1.00E-06	7.67	0.0000001*
	RAW	3.53	0.00549*	3.73	0.00223	4.57	0.0001497*
RF	LOC+TIME	2.82	0.01805*	3.76	0.002115	2.62	0.01573*
	MI	14.15	0.0000001*	7.16	4.90E-06	4.85	0.0000759*
	RAW	2.96	0.0144*	3.86	0.001734	4.15	0.0004139*

evident (Fig. 9). Among non-VLAIR approaches, LOC+TIME representations are the best, and the MI sensor data representations the worst.

Looking at individual datasets, VLAIR R+DMT+BC achieves the best scores for each datasets. Table 5 shows the results of one-sided paired Welch's t-tests (with $\alpha = 0.05$) comparing the F1-scores on each activity between the best VLAIR technique (R+DMT+BC with CNN+LSTM and animation input) and all the other techniques in non-VLAIR groups.

7.4. Execution times

Training times are expectedly large for deep learning models. For the most accurate VLAIR model the per-epoch training time is 351 s, 217 s and 558 s for Twor, Milan, and Aruba respectively. The average total training time is 9 h, 6 h, 17 h respectively. This is larger than the non-image-based deep learning models (5s, 6s and 12 s per epoch; 0.2 h, 0.2 h and 0.6 h total for the LSTM). As expected, total training times for classical models, which tend to be computationally cheaper, are much lower (Random Forests, the best of these, had training times of 120 s, 120 s and 720 s for the three datasets respectively). In terms of comparing classification time, for the most accurate VLAIR model the average per-image classification time was 0.0006s, and 0.0002s for the non-VLAIR model across all three datasets. Per-image classification time for the best classical model is 0.0000003s on average, also much shorter than with the deep models.

8. Discussion

Our tests comparing different visualizations show that there is some variation in accuracy across visualizations. Unsurprisingly, the richest and most sophisticated visualization resulted in the best results. Specifically, some visualization techniques such as avoiding overlap between transition traces through Bezier curves with variable curvatures seem to result in a substantial improvement. Interestingly, this is a mapping that we have developed when realizing, as viewers of example visualizations, that simple straight traces would occlude important information about the temporal relationships of multiple sequential activation of sensors.

In our tests, the best and the worst mappings are 17 to 19 percent points apart in the F1-macro scores, with more modest differences between 1 and 10 percent points in F1-micro scores. This suggests that the design of visualizations can have a substantial effect on the performance, yet even the simplest visualizations showed performance comparable to non-VLAIR approaches. Our results shows that a competent, but simple, visualization approach will achieve comparable results to non-VLAIR approaches, but to get a boost on performance it is necessary to design more sophisticated visualizations.

The results also indicate that the animation data input, coupled with a temporal architecture, results in better classification accuracy. There is an interesting parallelism here with visualization for humans: dynamic phenomena is often represented with visualizations that split temporal states into many sequential representations (small multiples), animations, or even interactive time slices (Tversky et al., 2002; Lee et al., 2020). Note that it is not strictly necessary to encode time aspects in this way (e.g., some of our own visualizations already encode the time dimension as color, and many visualizations encode time as space; e.g., timelines), yet the architectures that have explicit temporal components (LSTM) seem to be able to take better advantage of this information.

Surprisingly, the attention mechanisms that we have tested do not provide any obvious benefits in our experiments. This might be because our particular set of visualizations contains only basic connections and shapes that trace sensor activation during short periods of time. Indiscriminate extraction of local features at frame-level might be counterproductive because subregions of interest may not always be relevant to the activity at hand.

The key finding from our experiment is that, when comparing VLAIR and non-VLAIR approaches, the best VLAIR approaches offer the best classification accuracy. This is despite our best efforts to find architectures that are comparable in sophistication and that cover a broad set of non-visual representational forms. The increase in accuracy due to the use of our top VLAIR approach, compared to the best non-VLAIR approach is quite substantial. Accuracy increases of between 14 and 26 percentual points in F1-macro measures and between 1 percentual point and 8 percentual points in F1-micro measures can easily justify the additional computational cost of a deep learning approach in many applications. Nevertheless, if computation time (during training or for each individual classification) is a more important concern, then some classical ML approaches do have an advantage.

The best VLAIR approaches seem to be particularly good at picking up infrequent activity classes that many non-VLAIR approaches do not detect at all. For example, VLAIR correctly classifies the relatively infrequent (0.27%) Morning_Meds of the Milan dataset about half the time, whereas non-VLAIR approaches almost never detect these (see Fig. 11).

The VLAIR approaches seemed also to be able to better distinguish the same activity when it was performed by two different

Fig. 10. Comparison of RT+DMT+BC visualizations on R1 (top row) and R2 (bottom row) eating activities. The examples show that the visual encodings of time and movement help to separate activities that share similar sensor patterns.

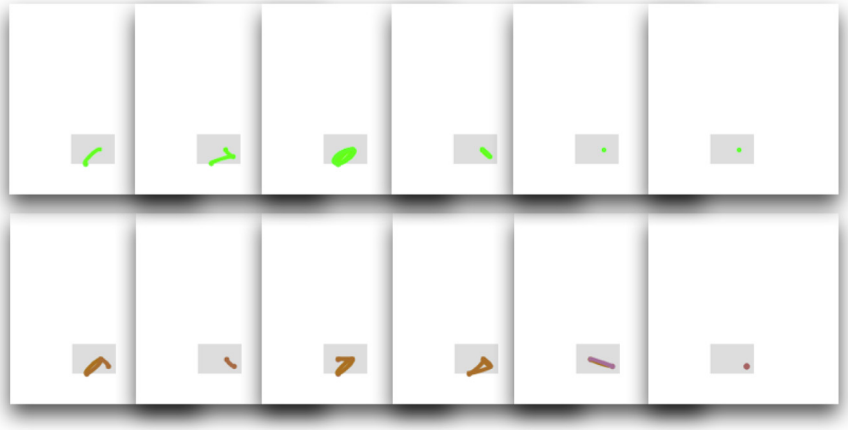


Fig. 11. VLAIR RT+DMT+BC images on a pair of activities that have similar patterns (Milan dataset). The baseline approaches misclassify 'Morning_Meds'(first row) as 'Kitchen_Activity'(second row) but VLAIR RT+DMT+BC can better distinguish them.

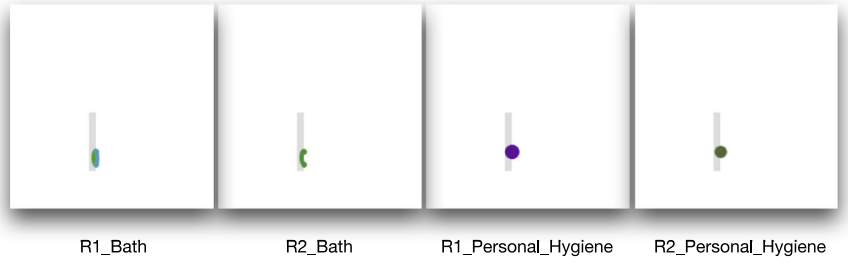


Fig. 12. Examples of VLAIR RT+DMT+BC images of activities that result in similar patterns in the Twor dataset. The converted images of 'Bath' and 'Personal_Hygiene' are almost identical, since they happen in the same place (bathroom) and usually at the same time for most of the daily routine. There is not much difference in their converted images on 'Bath' and 'Personal_Hygiene', but subtle difference between users on activation patterns (circles) and the users' daily schedules (trace color).

residents in the Tokyo dataset (see Fig. 10). The figure hints at how the color encoding of time, the size of circle-nodes and the thickness of lines could expose this information efficiently. Unfortunately, the opposite could also be true; i.e., there might be patterns that, due to their similarity in spatial and visual terms, are visually obscured despite having statistically differentiable characteristics (Fig. 12 might be one of these).

As to the reasons why VLAIR approaches might have shown these advantages we can only speculate. It is possible that the spatialization of the data allows the DNN to pick up patterns at a more superficial level, patterns that could require much deeper networks when analyzing data that is not previously spatialized. We also considered the possibility that, as Chen (2018) argues, that we can leverage existing sophisticated models of vision pre-trained on large datasets (e.g.,VGG16 (Simonyan and Zisserman, 2014), *ResNet* (He et al., 2016), *MobileNet* (Howard et al., 2017), *DenseNet* (Huang et al., 2017) and *InceptionV3* (Szegedy et al., 2016)) to achieve better accuracy. This is not the case for our experiments, in which we do not use this kind of transfer learning. In fact, we have tried this approach (we used VGG16) and have found that using a relatively simple CNN network trained from scratch provides better results. However, it is possible that

more sophisticated models could deliver better performance. Specifically, models developed to recognize objects in computer-generated graphics (rather than natural scenes from photographs) could be particularly useful.

Finally, it is worth highlighting that the process of coming up with visual mappings for a given dataset can be considered a form of feature selection, and hence encodes some level of expert knowledge about the data as well as about the characteristics of visual classification.

8.1. Lessons learned for the design of VLAIR mappings

Since we have carried out the experiments described in Section 6 we have been experimenting, ourselves and through student projects, with applications of the VLAIR technique on binary sensor data for human activity recognition. Here we advance some practical and anecdotal experience gathered about how to design visualization mappings for classification and regression through CV pipelines. This might support those applying VLAIR to start from a position of advantage.

- Color encodings, especially if they are subtle, are less effective than shapes or traces.
- Shapes, traces, and lines seem to work better than points and isolated circles.
- With a canvas that is so limited in space, dealing with data with high dynamic range requires careful thought.
- Representations of time-based data through sequences seem to work better than other—more abstract—encodings, such as color or thickness.

Notice that some of these might be consequences of the image representations and architectures that we have chosen for the CV pipeline; for example, color is split into three channels which might take more than three layers to integrate meaningfully.

8.2. Limitations

Our set of experiments covers only a particular application with one type of data. Although we tested three datasets it is impossible to come to a definitive conclusion about whether a majority, many, or only specific classification tasks can benefit from the VLAIR approach. Providing strong evidence for VLAIR's advantage would require a very large number of experiments over many different types of data, even if we only consider supervised classification tasks. We think that it is unrealistic to require this kind of evidence at this stage and that cautiously advocating for the use of the technique is granted by the promising results shown in our study. There are also several potential reasons beyond improved accuracy that might be attractive to practitioners; we discuss those in the next Section.

We are also aware that our coverage of comparison learning algorithms is not comprehensive. Despite our best efforts, we cannot rule out that specific non-visual alternative transformations of the data and other learning algorithms, including other deep learning pipelines with specific parameters and meta-parameters (e.g., more layers) might not outperform VLAIR approaches.

8.3. Potential opportunities for better interpretability

Intrinsic to the VLAIR process is the generation of visualizations from data. If these visualizations can also be efficiently perceived by humans they open an opportunity for people to find common ground with the machine learning algorithms. This, in turn, can help interpretability of the model and offer opportunities for debugging. Consider an example from our application scenario: if a model often confuses two distinct categories such as taking the morning medicines and regular kitchen activity (see Fig. 11) a technician could visually compare the patterns in the visualizations of both groups and might be able to detect the source of the misclassification. Perhaps an additional sensor in the kitchen could help discriminate the two activities, or perhaps a different visual mapping could make the two groups more visually distinctive.

In essence, the shared common intermediate representations might enable humans to use their own visual system as an accessible and very familiar surrogate of the computational model (what Adadi and Berrada classify as post-hoc interpretability (Adadi and Berrada, 2018), except that the explanatory model does not have to be designed because it already comes standard for most humans). It is important to highlight also that using VLAIR does not preclude other interpretability or explainability techniques already in use. In fact, it might enable the use of techniques from CV that are not yet applicable in non-image data. For example, Selvaraju et al.'s Grad-CAM could highlight the areas of the visualization that dominate a particular classification decision hence helping the viewer provide visual explanations for issues (for other such methods see, e.g., Montavon et al. 2018).

This leads us to three main open research questions: *Is human visual observation of samples or augmented visual explanations sufficiently accessible for humans to generate useful interpretation of the model outputs?*, *Is the human visual system similar enough to the generated CV model to function as a useful surrogate?*, *What type of visualization mappings are good for both human interpretation and machine learning accuracy?*.

9. Conclusion and future work

This paper presents and discusses VLAIR, an approach to applying deep learning that leverages the strength of data visualization in making inherent features explicit for deep learning models. We have presented the approach, reviewed the existing work that uses related principles, and discussed its potential advantages especially in encoding and uncovering spatial and temporal patterns that are not immediately visible in raw sensor data. In a comparison with baselines from classical machine learning and deep learning algorithms we have found that VLAIR can significantly boost classification accuracy in a human activity recognition example.

There are several additional avenues of research complementary to finding answers to the open questions raised above. First, it is possible to explore whether more sophisticated mappings from the vast visualization literature could outperform the relatively simple ones that we tested. Second, it is important to validate with user studies our own experience of the interpretation of visualizations to iteratively design better mappings. Third, a more customized CV pipeline design might provide a better foundation for VLAIR approaches. Fourth, we might be able to leverage differences between machine learning and computer vision algorithms and the human visual system to provide perception that goes beyond that of a human.

CRediT authorship contribution statement

Ai Jiang: Conceptualization, Methodology, Software, Validation, Investigation, Formal analysis, Writing – original draft, Writing – review & editing. **Miguel A. Nacenta:** Writing – original draft, Writing – review & editing. **Juan Ye:** Writing – original draft, Writing – review & editing, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We thank the China Scholarship Council (CSC) for financially supporting my PhD study at University of St Andrews, UK, and NSERC Discovery Grant 2020-04401 (Miguel Nacenta). We also thank Paria Naghavi for comments on manuscript.

Ethical approval

This study does not contain any studies with human or animal subjects performed by any of the authors. All data used in the study are taken from public databases that were published in the past.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.visinf.2022.05.001.

References

Adadi, A., Berrada, M., 2018. Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). IEEE Access 6, 52138–52160, Conference Name: IEEE Access.

Ahmad, Z., Khan, N., 2018. Towards improved human action recognition using convolutional neural networks and multimodal fusion of depth and inertial sensor data. In: 2018 IEEE International Symposium on Multimedia. ISM, pp. 223–230.

Alberdi, A., Weakley, A., Schmitter-Edgecombe, M., Cook, D.J., Aztiria, A., Basarab, A., Barrenechea, M., 2018. Smart home-based prediction of multidomain symptoms related to alzheimer's disease. IEEE J. Biomed. Health Inf. 22 (6), 1720–1731.

Aminikhanghahi, S., Wang, T., Cook, D.J., 2018. Real-time change point detection with application to smart home time series data. IEEE Trans. Knowl. Data Eng. 31 (5), 1010–1023.

Bianchi, V., Bassoli, M., Lombardo, G., Fornacciari, P., Mordonini, M., De Munari, I., 2019. IoT wearable sensor and deep learning: An integrated approach for personalized human activity recognition in a smart home environment. IEEE Internet Things J. 6 (5), 8553–8562.

Chegini, M., Bernard, J., Berger, P., Sourin, A., Andrews, K., Schreck, T., 2019. Interactive labelling of a multivariate dataset for supervised machine learning using linked visualisations, clustering, and active learning. Vis. Inf. 3 (1), 9–17, Proceedings of PacificVAST 2019.

Chen, L., 2018. Deep transfer learning for static malware classification. CoRR, abs/1812.07606 arXiv:1812.07606.

Cook, D.J., Crandall, A.S., Thomas, B.L., Krishnan, N.C., 2013. CASAS: A smart home in a box. Computer 46 (7), 62–69.

Cook, D.J., Schmitter-Edgecombe, M., 2009. Assessing the quality of activities in a smart environment. Methods Inf. Med. 48, 480–485.

Fan, C., Matković, K., Hauser, H., 2021. Sketch-based fast and accurate querying of time series using parameter-sharing LSTM networks. IEEE Trans. Vis. Comput. Graphics 27 (12), 4495–4506.

Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L., Muller, P.-A., 2019. Deep learning for time series classification: a review. Data Min. Knowl. Discov. 33 (4), 917–963.

Feuz, K.D., Cook, D.J., 2017. Collegial activity learning between heterogeneous sensors. Knowl. Inf. Syst. 53 (2), 337–364.

Fukushima, K., 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol. Cybernet. 36 (4), 193–202.

Graves, A., Mohamed, A.-r., Hinton, G., 2013. Speech recognition with deep recurrent neural networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, pp. 6645–6649.

Ha, S., Yun, J.-M., Choi, S., 2015. Multi-modal convolutional neural networks for activity recognition. In: 2015 IEEE International Conference on Systems, Man, and Cybernetics. IEEE, Hong Kong, pp. 3017–3022.

Hammerla, N.Y., Halloran, S., Plötz, T., 2016. Deep, convolutional, and recurrent models for human activity recognition using wearables. arXiv preprint arXiv:1604.08880.

Hatami, N., Gavet, Y., Debayle, J., 2018. Classification of time-series images using deep convolutional neural networks. In: Tenth International Conference on Machine Vision, vol. 10696. ICMV 2017, International Society for Optics and Photonics, p. 106960Y.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, Las Vegas, NV, USA, pp. 770–778.

Heer, J., Bostock, M., 2010. Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, Atlanta, Georgia, USA, pp. 203–212.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv arXiv:1704.04861.

Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 2261–2269.

Jiang, A., Nacenta, M.A., Terzic, K., Ye, J., 2020. Visualization as intermediate representations (VLAIR) for human activity recognition. In: Proceedings of PervasiveHealth'2020.

Jiang, W., Yin, Z., 2015. Human activity recognition using wearable sensors by deep convolutional neural networks. In: Proceedings of the 23rd ACM International Conference on Multimedia. ACM, pp. 1307–1310.

van Kasteren, T., Englebienne, G., Kröse, B., 2011. Human activity recognition from wireless sensor network data: Benchmark and software. In: Activity Recognition in Pervasive Intelligent Environments. Atlantis Press, Paris, pp. 165–186.

Krishnan, N.C., Cook, D.J., 2014. Activity recognition on streaming sensor data. Pervasive Mob. Comput. 10, 138–154.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems. pp. 1097–1105.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521 (7553), 436–444.

LeCun, Y., Kavukcuoglu, K., Farabet, C., 2010. Convolutional networks and applications in vision. In: Proceedings of 2010 IEEE International Symposium on Circuits and Systems. IEEE, pp. 253–256.

Lee, A., Archambault, D., Nacenta, M.A., 2020. The effectiveness of interactive visualization techniques for time navigation of dynamic graphs on large displays. arXiv:2008.12747 [Cs].

Li, Q., Wu, Z., Xu, P., Qu, H., Ma, X., 2018. A multi-phased co-design of an interactive analytics system for MOBA game occurrences. In: Proceedings of the 2018 Designing Interactive Systems Conference. In: DIS '18, Association for Computing Machinery, New York, NY, USA, pp. 1321–1332.

Liu, H., Chen, X., Wang, Y., Zhang, B., Chen, Y., Zhao, Y., Zhou, F., 2021. Visualization and visual analysis of vessel trajectory data: A survey. Vis. Inf. 5 (4), 1–10.

Logan, B., Healey, J., Philipose, M., Tapia, E.M., Intille, S., 2007. A long-term evaluation of sensing modalities for activity recognition. In: Krumm, J., Abowd, G.D., Seneviratne, A., Strang, T. (Eds.), UbiComp 2007. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 483–500.

Luong, T., Pham, H., Manning, C.D., 2015. Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Lisbon, Portugal, pp. 1412–1421, URL https://www.aclweb.org/anthology/D15-1166.

Manovich, L., 2011. What is visualisation? Vis. Stud. 26 (1), 36–49.

Montavon, G., Samek, W., Müller, K.-R., 2018. Methods for interpreting and understanding deep neural networks. Digit. Signal Process. 73, 1–15.

Morales, F.J.O.n., Roggen, D., 2016. Deep convolutional feature transfer across mobile activity recognition domains, sensor modalities and locations. In: ISWC '16. pp. 92–99.

Patel, A., Shah, J., 2019. Sensor-based activity recognition in the context of ambient assisted living systems: A review. J. Ambient Intell. Smart Environ. 11 (4), 301–322.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.

Pourbabaee, B., Roshtkhari, M.J., Khorasani, K., 2017. Deep convolutional neural networks and learning ECG features for screening paroxysmal atrial fibrillation patients. IEEE Trans. Syst. Man Cybern. Syst. (99), 1–10.

Radu, V., Tong, C., Bhattacharya, S., Lane, N.D., Mascolo, C., Marina, M.K., Kawsar, F., 2018. Multimodal deep learning for activity and context recognition. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 1 (4), 157:1–157:27.

Ravi, D., Wong, C., Lo, B., Yang, G.-Z., 2016. Deep learning for human activity recognition: A resource efficient implementation on low-power devices. In: BSN '16. IEEE, pp. 71–76.

Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Singh, M.S., Pondenkandath, V., Zhou, B., Lukowicz, P., Liwickit, M., 2017. Transforming sensor data to the image domain for deep learningâan application to footstep detection. In: 2017 International Joint Conference on Neural Networks. IJCNN, IEEE, pp. 2665–2672.

Sprint, G., Cook, D., Weeks, D., Dahmen, J., La Fleur, A., 2017. Analyzing sensor-based time series data to track changes in physical activity during inpatient rehabilitation. Sensors 17 (10).

Stoiber, C., Ceneda, D., Wagner, M., Schetinger, V., Gschwandtner, T., Streit, M., Miksch, S., Aigner, W., 2022. Perspectives of visualization onboarding and guidance in VA. Vis. Inf..

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 2818–2826.

Thomaz, E., Bettadapura, V., Reyes, G., Sandesh, M., Schindler, G., Plötz, T., Abowd, G.D., Essa, I., 2012. Recognizing water-based activities in the home through infrastructure-mediated sensing. In: Proceedings of the 2012 ACM Conference on Ubiquitous Computing. In: UbiComp '12, Association for Computing Machinery, New York, NY, USA, p. 85?94.

Tversky, B., Morrison, J.B., Betrancourt, M., 2002. Animation: can it facilitate? Int. J. Hum.-Comput. Stud. 57 (4), 247–262.

Wang, J., Chen, Y., Hao, S., Peng, X., Hu, L., 2019. Deep learning for sensor-based activity recognition: A survey. Pattern Recognit. Lett. 119, 3–11, Deep Learning for Pattern Recognition.

Wang, X., Girshick, R., Gupta, A., He, K., 2018. Non-local neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7794–7803.

Wang, Z., Oates, T., 2015. Encoding time series as images for visual inspection and classification using tiled convolutional neural networks. In: Workshops At the Twenty-Ninth AAAI Conference on Artificial Intelligence, vol. 1.

Wang, J., Zhang, X., Gao, Q., Yue, H., Wang, H., 2017. Device-free wireless localization and activity recognition: A deep learning approach. IEEE Trans. Veh. Technol. 66 (7), 6258–6267.

Wu, Z., Yao, T., Fu, Y., Jiang, Y.-G., 2017. Deep learning for video classification and captioning. In: Frontiers of Multimedia Research. pp. 3–29.

Xiong, G., Fu, Q., Fu, H., Zhou, B., Luo, G., Deng, Z., 2021. Motion planning for convertible indoor scene layout design. IEEE Trans. Vis. Comput. Graphics 27 (12), 4413–4424.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y., 2015. Show, attend and tell: Neural image caption generation with visual attention. In: International Conference on Machine Learning. pp. 2048–2057.

Ye, J., Stevenson, G., Dobson, S., 2015. KCAR: A knowledge-driven approach for concurrent activity recognition. Pervasive Mob. Comput. 19, 47–70.

Zeng, W., Lin, C., Lin, J., Jiang, J., Xia, J., Turkay, C., Chen, W., 2020. Revisiting the modifiable areal unit problem in deep traffic prediction with visual analytics. IEEE Trans. Vis. Comput. Graphics 27 (2), 839–848.

Zeng, M., Nguyen, L.T., Yu, B., Mengshoel, O.J., Zhu, J., Wu, P., Zhang, J., 2014. Convolutional neural networks for human activity recognition using mobile sensors. In: 6th International Conference on Mobile Computing, Applications and Services. IEEE, pp. 197–205.

Zhang, J., Zheng, Y., Qi, D., 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In: Thirty-First AAAI Conference on Artificial Intelligence.