Internship Report

Relevant Frame Selection for Human Action Recognition

Matthieu Medeng Essia

M1 Data Science

Research Team : CERI SN, HIDE

Supervisors : Omar Ikne, Benjamin Allaert

Academic Supervisor : Pierre Tirilly

University of Lille

IMT North Europe

Villeneuve-d'Ascq, France

May 9th - August 11th

Abstract Relevant Frame Selection for video classification or action recognition is actually a open subject in Computer Vision because of the variable size of the videos in datasets. One common method to cope with it is padding each sequence to the length of the longest one, which is quite expensive in terms of complexity. Although many ways have already been explored by researchers throughout the Literature to reduce the number of frames per sequence to the strict necessary ones, there is still a lot to do to achieve high performances video classification with a minimal number of relevant frames in a sequence of images. The internship project aims at exploring existing Frame Selection methods and proposing a less usual one that will be based on attention calculated on sequences' frames to retain those with the most interesting attention.

Contents

List of Figures

List of Tables

1 Internship Context

1.1 The Research Centre

At IMT North Europe, the Centre for Education, Research and Innovation (CERI) Digital Systems (SN) covers a wide disciplinary field linked to constrained systems (the Internet of Objects, robotics), Humans (and in particular their interactions with the digital world) or even complex systems through the prism of Artificial Intelligence and Automation. The 34 lecturer-researchers and 6 engineers at CERI are able to cover all teaching fields in the field of digital sciences and technologies (Telecoms, Networks, Systems, Data, Artificial Intelligence, Applications, Cybersecurity, etc.).

It is structured around 3 research groups: ARTS (Autonomous ResilienT Systems), HIDE (Human, Interaction, DEcision) and McLEOD (Modelling and Control of Complex systems in Large Environments requiring Optimized Decision).

In addition, there is the PPI (Innovation Platform), which supports teacher-researchers in carrying out their prototypes and promoting these to companies. The PPI is also able to provide support services to other CERIs as well as companies.

In this context was proposed the internship mission with the research team of CERI SN whose activities mainly focus on interaction between new technologies and human society.

In fact, the mission consisting in performing relevant Frame Selection in Human Motion Analysis and Action Recognition domain, the research project will be part of the work of the research group HIDE.

1.2 The Research Group

The research group HIDE aims at daily interacting with computer systems, such as smartphones and computers, but also transport, information, surveillance, communication systems, and much more. For greater ease and fluidity, the interactions between humans and these systems need to be simplified. The latter must learn to adapt to humans and offer them the means to express their needs more effectively and more naturally via new interaction tools. In return, they must produce more intelligible and reliable results, thanks

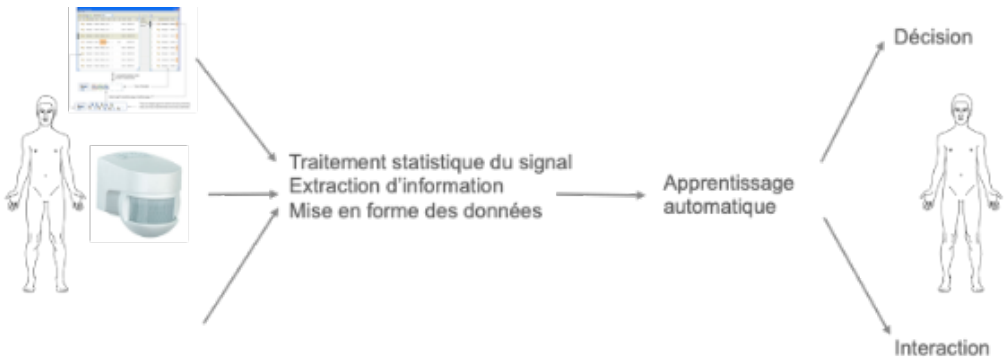


Figure 1: An overview of HIDE's missions

to the automatic extraction of relevant information, which will help humans to reason better and make the best decisions in complex situations (see Fig. 1).

The objective of this group is to develop models and methods for digital simulation, machine learning or even decision making to help humans reason better, improve their interaction with their environment and better optimise certain processes.

Different fields of application are targeted, including the following:

- **Intelligent Tutoring System:** in both software engineering and data analysis, the group's contribution is based on a better understanding of the sequences of the processes in order to help the person.
- **Health:** in this field, the independence or the well-being of the person, in particular one who lives alone, is a focus of interest in order to detect short or long term problems.
- **Interaction:** the objective is to promote the use of hand gestures and eye tracking as a complete and natural tool for human-machine interactions, particularly in virtual environments in virtual reality or in augmented reality.
- **Transport:** behaviour modelling can be used to analyse road infrastructure. To do this, multi-sensor analyses (including deep learning on the image or LIDAR) provide the use of infrastructures by users to simulate modified versions.
- **Information/Journalism:** mostly related to helping analyse the reliability of sources.

- **Telecommunications:** the objective is to allow any user to transmit their data at a sufficient rate but so as to maximise the capacity of the overall communication network.
- **Design and maintenance environments for CPSoS and Interactive Machine Learning systems.**

As it will be mentioned in the next part of the report, the final objective of the internship mission is to provide applications that will serve the cause of a national project in transport domain, namely focusing on security of human being in smart transports, a key point of Human Interaction with such vehicles.

2 Introduction

The main purpose of the proposed internship is related to the ECOTRAIN project in which the research team of CERI SN at IMT North Europe takes a major part, that is, detecting obstacles to anticipate train's braking and stop. More specifically, the internship focuses itself on vulnerable users (pedestrians, cars, ...) through detection and anticipation of their behaviour. This is quite an ambitious project, financially supported by ADEME, the french agency for environment and energy gestion, that aims at facilitating the transport of people in rural areas (see Fig. 2). That said, to achieve autonomy in anticipating human behaviour with an autonomous train, the internship project will be dealing with Human Action Recognition, which is a widely spread thematic in Deep Learning and Computer Vision.

(a) Autonomous electric train.

IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

IMT Mines Albi-Carmaux
École Mines-Télécom

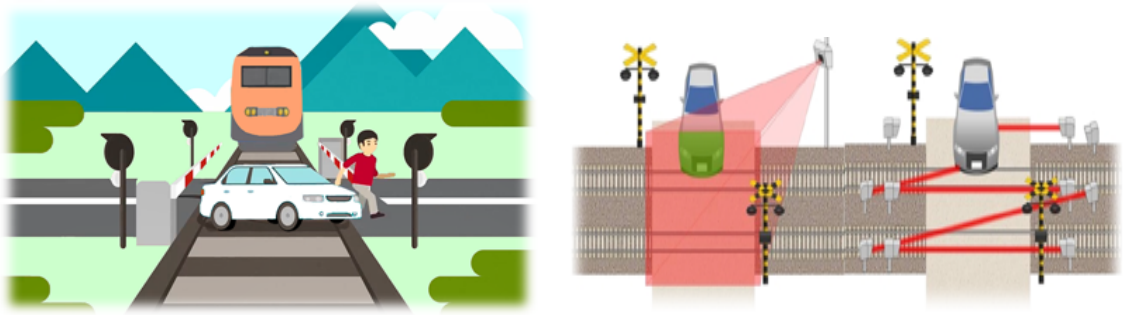
(b) Academic partners.

SYNTONY GNSS
CLEARSY Safety Solutions Designer
Socofer FRANCE
Groupe Stratiforme Compreforme

(c) Industrial partners.

Figure 2: The project's purpose and its main actors.

The ECOTRAIN project consists of two parts, that are, an embedded one in the train on one hand, and a fixed part for the infrastructures on the other. In the internship mission, we are focusing on the second one as it will imply a low energetic cost and a fast inference. More precisely, one intends to make the shuttle communicate with poles equipped with AI and a camera recording what is happening at crossroads where pedestrians and other vehicles are likelier to encounter the train. Entering a perimeter of 1000 meters, the shuttle, whose velocity is 70 km/h, shall interrogate the pole about the possibility of passing through or



(a) Risk analysis of vulnerable users presence. (b) Pole's detection role.

Figure 3: Shuttle-pole interaction in risk analysis.

starting to stop. In order to respond, the pole shall transmit the decision it took according to what its camera saw (see Fig. 3). However, due to governmental constraints, those poles will have to be energetically autonomous, which raises an issue: finding a way to minimize each pole energetic consumption, which is not trivial since the sequences filmed by the camera last for hours. Therefore, when processing sequences of frames to detect human presence on the railway, poles must be able to take judicious decisions using as few frames as possible to reduce their consumption and optimize the processing time. This is the objective of the project: developing a way to extract key frames of very long sequences.

2.1 Problematic

Nevertheless, two major problems emerge from this context: the choice of architectures that are suited to help analyze human motion and the most relevant way to select equally partitioned sequences of frames to feed the models with.

What concerns the choice of the model, on one hand, it has been proved in the Literature that performing Human Action Recognition using 3D Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) turns out to give interesting results. On the other hand, nowadays' techniques involve more efficient models using Reinforcement Learning methods such as attention modules in Transformers.

As for Frame Selection, the core of the internship project, the task appears to be more complex. Let us consider the case of video surveillance in danger detection. Cameras are usually launched to record sequences during hours but only a few seconds are sufficient to

detect an anomaly. As a result, the number of frames required to recognize a particular event is far shorter than the duration of the whole video. With this in mind, to achieve efficient video recognition, the team's mission will be to explore how to automatically adjust computation within a network on a per-video basis such that —conditioned on different input videos, a small number of informative frames are selected to produce correct predictions. However, this is a particularly challenging problem, since videos are generally weakly-labeled for classification tasks, one annotation for a whole sequence, and there is no supervision informing which frames are important. Therefore, it is unclear how to effectively explore temporal information over time to choose which frames to use, and how to encode temporal dynamics in these selected frames [9].

2.2 Objectives

The Research Team -the supervisors in particular- have already explored a large part of Deep Learning and Computer Vision applications when it comes to the analysis of human behaviour through video streams. This topic even plays a key role in the PhD thesis they are leading. So one point of this mission is also to provide a pipeline that may widen the horizon of the thesis.

Besides, considering the fact that frame selection in Human Action Recognition is actually an open subject in Computer Vision, another objective of the internship is to explore the state-of-the-art techniques in the Literature (handcrafted methods or learning-based approaches) that allow to find solutions to determining which frames are important in a sequence.

Furthermore, this mission is of course an occasion to master the notions that have been taught in the M1 course and anticipate the notions coming next in M2's. For instance, one will be able to achieve data preprocessing, video transformation into frames, Neural Networks [2, 8, 4] and Transformers [7, 5] implementation, optical flow [1], joints and landmarks computation [3], and so on.

2.3 Workplan

In the sequel of the report will be presented the Research aspects that are at stake for the project in Section 3 , namely the role played by Frame Selection in the Literature as well as the impact of Reinforcement Learning in improving Frame Selection techniques.

Moreover, in Section 4 part will be discussed the solutions to the above mentioned problems, that are, the way of choosing relevant frames and the most interesting Deep Learning architectures for the experiments that have been led.

Right after, in Section 5 part will be introduced the dataset of interest, and given the implementation details for the best results obtained in the approach and the different raw data that have been exploited as well as their positive or negative influence on the results.

In the end, in Section 6 will be debated the relevance of the obtained results.

3 Related Work

The field of Action Recognition is wide and includes a large variety of sub problems, and families of methods. Here one tends to focus on two domains within action recognition that constitute the core of the subject: Frame Selection, and attention modules, which will help the selection in a more efficient way.

3.1 Frame Selection

Selecting important frames for action recognition is a relatively new area. Many approaches have successfully trained a Reinforcement Learning (RL) [4] agent approach that examines one frame at a time, to predict how many frames can be skipped. Among others, one might cite AdaFrame [9], which leverages RL, in combination with a Long Short-Term Memory cell (LSTM) [4] that is augmented with memory that helps provide context information for selecting frames to use. Given a frame, it generates a prediction of the action class and it decides which frame to observe next and computes the expected reward of seeing more frames. The selection is performed based on an attention system from the computed weights returned by the LSTM (see Fig. 4).

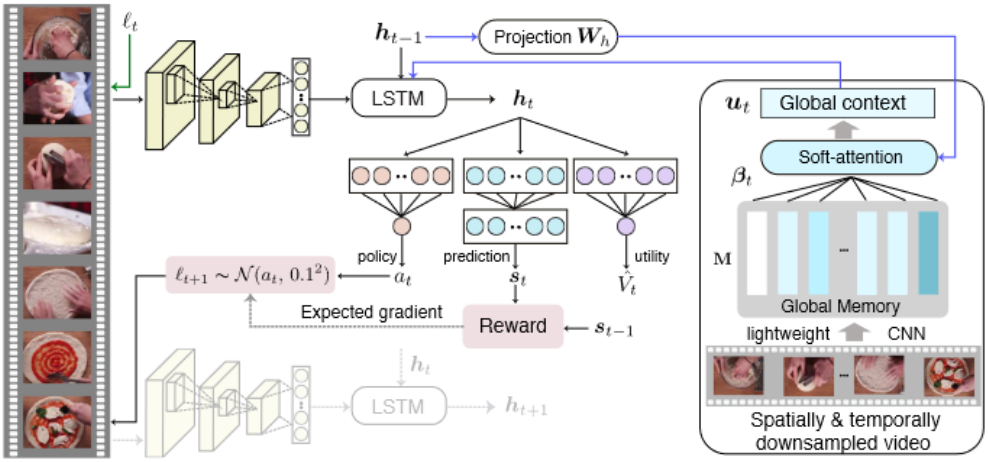


Figure 4: AdaFrame Model Architecture [9].

A first attempt to handle the proposed topic was at first to use handcrafted frame selection methods existing in the literature. The most commonly used technique is the padding, which consists in adding black images (matrices of zeros) to a sequence so that its length (number

of images) can match other sequences. Those images can be added either to the left or to the right (see Fig. 5). For instance, if one works with a set of sequences whose maximal length is a sequence of 384 frames, all other sequences with inferior length will be completed by black images until the number of frames will reach 384. That means, if a sequence possesses 100 frames within itself, 284 black images will be added to it. The main advantage with this technique is that one keeps the whole information over each sequence. However, using this method increases the complexity in time and space, which is problematic considering the final goal, that is achieving low energetic consumption.

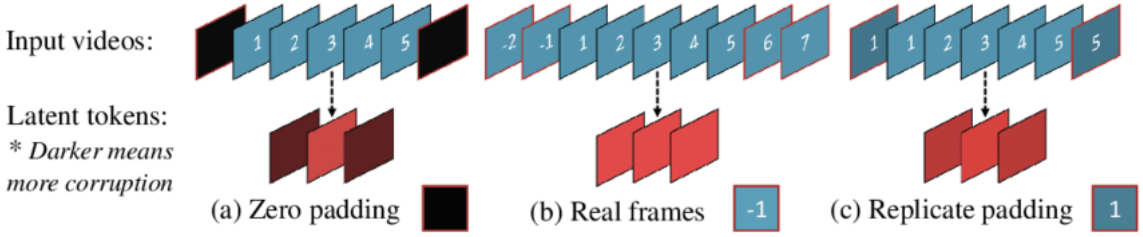


Figure 5: Padding Frame Selection.

Another interesting technique consists in computing the magnitude of motion between each frame and its successor and keeping the desired number of frames with enough amplitude between them (Motion Guided) [10]. This one is very practical in the way it allows to considerably reduce the number of frames per sequence while keeping the most significant frames in terms of amplitude, which optimizes the learning process since the model will focus on the main parts of the motion. One disadvantage with that technique is that one might erase some important information from the video, namely temporal information, which might make it difficult to classify similar actions whose fundamental difference resides in the time it takes to perform them. For example, when one wants to make a model distinguish "running" from "jogging".

Another method, much simpler consists in just taking the first k frames of a sequence, padding those which does not have enough images and ignoring the frames of those exceeding the required number. One advantage is that one can significantly reduce the number of frames per sequence and still keep enough of them to help a model recognize motion. But the main side effect of this technique is the fact that it only keeps the first scenes in the action, which means that if not enough images have been taken, one could easily miss the middle part

and the end of motion, which makes the model miss a lot of information and might end in under-fitting.

A second approach was the use of Vision Transformers [7, 5], that are recent and very efficient Deep Learning models used for Action Recognition, to classify the sequences and retrieve the computed attention weights to keep for each sequence the frames that have received more attention than the others. The main advantage with this method is the reliability of selected frames, which ensures that the images kept over the sequence are the ones which describe motion the best and will optimize the learning process, while keeping only few frames. The main disadvantage will also be the lack of temporality as there exist actions that need to be studied with their execution time in order to distinguish them in spite of their similarities.

After selection, the obtained frames are given to train a 3D CNN [2], RNN [8] or Transformer [7] to classify motion.

3.2 Classification Models

A large number of Action Recognition methods have been proposed in the literature. Among them, many methods employ the Deep Learning models, including both the Convolutional Neural Network (CNN) and the Recurrent Neural Network (RNN). CNN-based methods usually summarizes the information from all frames into one image and then apply CNN for classification on this single image. On the contrary, RNN-based methods sequentially process the frames and classify the sequences after all the frames are given.

In Action Recognition field, RNN-based methods have been recently augmented by the incorporation of attention modules to explicitly model the observation that, for different actions, different features may show different degrees of importance in classification. An attention module is an input processing technique of Neural Networks. This mechanism helps Neural Networks solve complicated tasks by dividing them into smaller areas of attention and processing them sequentially. Just as the human brain solves a complex task by dividing it into simpler tasks and focusing on them one by one, the attention mechanism makes it possible for Neural Networks to handle intuitive and challenging tasks like translation and generating subtitles. The Neural Network focuses on specific aspects of a complex input until it categorises the entire dataset. The most recent Independent Recurrent Neural Network

(IndRNN) provides an effective solution to the gradient vanishing and exploding problem in training a multiple-layer RNN, which allows deep networks to be constructed and to learn long-term dependency. Specifically, preliminary experiments using multiple layers of the basic IndRNN have shown that better performance than LSTM-based networks on action recognition can be attained [3].

A Transformer is a model architecture that eschews recurrence and instead relies entirely on an attention mechanism to draw global dependencies between input and output. Before Transformers, the dominant sequence transduction models were based on complex Recurrent or Convolutional Neural Networks that include an encoder and a decoder. The Transformer also employs an encoder and a decoder, but removing recurrence in favor of attention mechanisms allows significantly more parallelization than methods like RNNs and CNNs [7].

4 Detailed Approach

This section will be devoted to explaining the Frame Selection techniques that have been explored and the model architectures that came along with them.

4.1 Frame Selection Techniques

In order to fit in the model, each sequence of frames must have the same length. Therefore, it becomes necessary to apply a method to relevantly choose the best frames to feed the model with.

The first idea was to pad all sequences to reach the longest sequence size either to the right or to the left (see Fig. 5). This method is often used to obtain quick results in image processing. However, there exists more elegant ways to play with sequence length while keeping significant information.

Therefore, another solution has been to compute the magnitude of motion between consecutive frames and fix a properly chosen sequence length to keep it as the number of frames to select according to the magnitude of motion between them and their successor. Naturally, the frames with maximal amplitude, i.e. those describing occurring motion the best, are the ones that get selected (see Fig. 6).

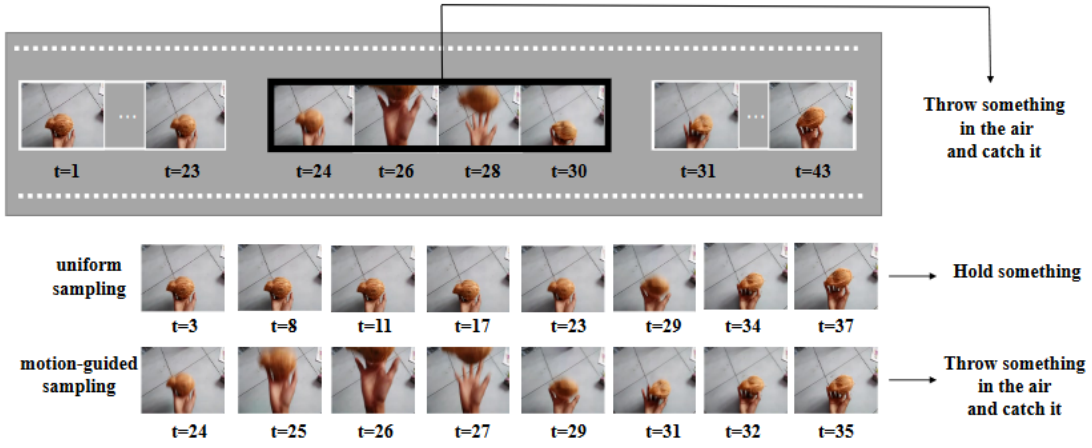


Figure 6: Motion Guided Frame Selection [10].

Although those approaches working with RGB raw data present a lot of advantages, other interesting options were to be explored, as optical flow and human skeleton joints, among

others.

In the team's project workspace, Python modules for computing optical flow were at disposition. Several algorithms such as Farneback [1], TVL1, Deep or PCA were used to do so. These modules even allow to manipulate optical flow in the form of polar coordinates (magnitude and angle), an image (height, width and channel) or directional vectors u and v (its base form). One just needed to define an algorithm to threshold the magnitudes of the optical flow computed in polar coordinates over the sequences by minoring the corresponding histogram's observed amplitudes to keep the most significant ones.

Concerning the computation of skeleton joints, using the Python library called "mediapipe", one created a module for posing the joints at their coordinates using mediapipe's detection algorithms for images. Each time when the raw data got modified, were the frames of each sequence transformed into joints or optical flow using those modules according to which form of data one wanted to feed the model with.

4.2 Model Architectures

In this section will be described the state-of-the-art approaches that have been involved in the research project due to their efficiency at classifying videos in Human Action Recognition domain.

4.2.1 3D Convolution Neural Networks

In 2D Convolutional Neural Networks, convolutions are enforced on the two dimensional maps extracted from the feature to enumerate the features available from the geometrical dimension. Now, 3D Convolutional Neural Networks aim at gauging the features from both temporal and spatial dimensions, which corresponds to the Human Action Recognition task where most of the existing datasets are made out of videos.

A common design scheme of CNNs is the number of feature maps which grows as the layers increase by developing the various multiple types of features from the available lower level of maps. The 3D convolution is obtained by convolving a 3D filter kernel by stacking multiple contiguous frames together to produce the 3D cube. Formally, the value at position (x, y, z) on the j^{th} feature map in the i^{th} layer is given by

$$v_{i,j}^{x,y,z} = \tanh\left(b_{ij} + \sum_{m}\sum_{a=0}^{A_i-1}\sum_{b=0}^{B_i-1}\sum_{c=0}^{C_i-1} w_{ijm}^{abc} v_{(i-1)m}^{(x+a)(y+b)(z+c)}\right) \quad (1)$$

where C_i is the 3D filter kernel size along the temporal dimension w_{ijm}^{abc} is the (a, b, c) is the feature map connected to the m^{th} value of the kernel in the previous layer. Here is an overview of the described architecture (see Fig. 7).

When it comes to CNNs, some hyper parameters have to be taken into account, namely the receptive field (R), the zero-padding (P), the stride length (S) and the volume dimensions (Width x Height x Depth or W x H x D). To compute the convolution layer neurons, one applies the formula $((W - F + 2.P)/S) + 1$. In the case of the presented 3D CNN, the input layer will be turned into $((120 - 11 + 0)/1) + 1 = 110$ features of output volume of $110 \times 110 \times 32$. So, the Width and Height parameters have been set to 120 for the input frame, $F = 11 \times 11 \times 32$ is the 3D filter depth, $P = 0$ is the zero-padding and $S = 1$ is the chosen stride.

In this 3D CNN, the two convolutional layers use kernel filters of size $11 \times 11 \times 32$ and $5 \times 5 \times 32$ respectively, and the two layers of max pooling use kernels of size SLM $\times 2 \times 2$ (where SLM is a multiple of the sequence length), progressively reducing the number of parameters and network computation, and also regulating the overfitting and then, a fully connected layer handles all features from the previous layer that are flattened into 6400 Dimensional feature vectors. The softmax layer consists of output units leading to the action classes [2].

4.2.2 Recurrent Neural Networks

A Recurrent Neural Network is a network that can understand sequences and time. In a Convolutional Neural Network, we were treating a single image every time. If we work on a video, we split the video into images and treat each image independently. In a Recurrent Neural Network, sequences are a set of time-related data.

RNN is a class of neural network that maintains internal hidden states to model the dynamic temporal behaviour of sequences with arbitrary lengths through directed cyclic connections between its units. It can be considered as a hidden Markov model extension that employs nonlinear transition function and is capable of modeling long term temporal

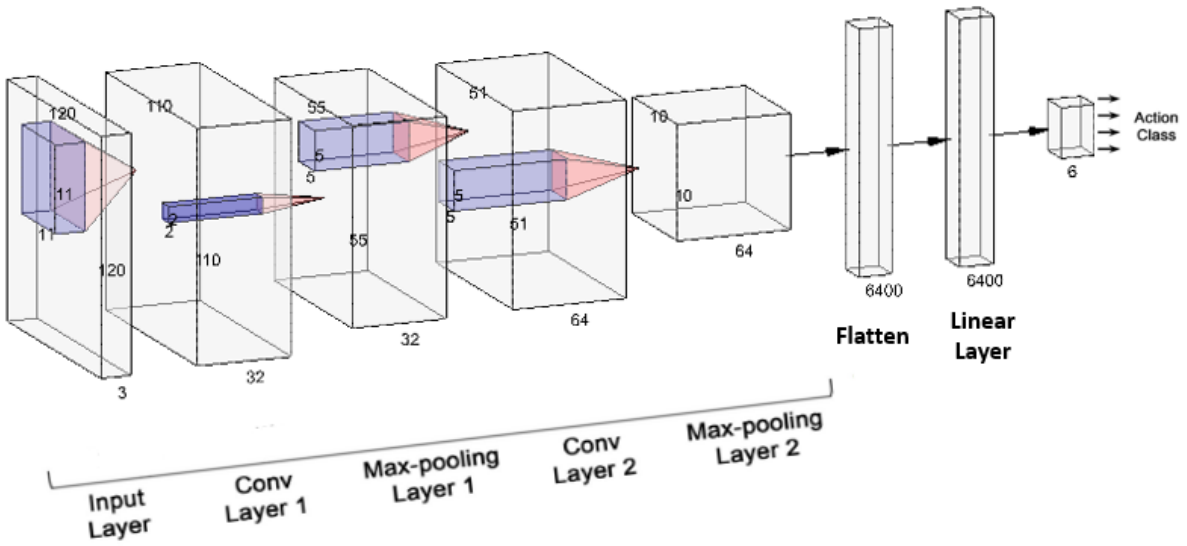


Figure 7: Overview of the 3D CNN architecture [2].

dependencies. LSTM extends RNN by adding three gates to an RNN neuron: a forget gate f to control whether to forget the current state; an input gate i to indicate if it should read the input; an output gate o to control whether to output the state. These gates enable LSTM to learn long-term dependency in a sequence, and make it is easier to optimize, because these gates help the input signal to effectively propagate through the recurrent hidden states $r(t)$ without affecting the output. LSTM also effectively deals with the gradient vanishing/exploding issues that commonly appear during RNN training [8].

$$x_t = \delta(U_r \cdot r(t-1) + U_w w_k(t)) \tag{2}$$

$$i_t = \delta(U_{i_r} r(t-1) + U_{i_w} w_k(t)) \tag{3}$$

$$f_t = \delta(U_{f_r} r(t-1) + U_{f_w} w_k(t)) \tag{4}$$

$$o_t = \delta(U_{o_r} r(t-1) + U_{o_w} w_k(t)) \tag{5}$$

$$r(t) = f_t \odot r(t-1) + i_t \odot x_t \tag{6}$$

$$o(t) = r(t) \odot o(t) \tag{7}$$

where $\delta(.)$ is an activation function, $\odot$ is the product with gate value, and various W matrices are learned parameters.

The proposed RNN model is a state-of-the-art one that has been proven to give good result, that is, the CNN-RNN framework for multi-label classification problem (see Fig. 8). The CNN part extracts semantic representations from images; the RNN part models image/label relationship and label dependency [8].

The probability of a prediction path can be computed by the RNN network. The image, label, and recurrent representations are projected to the same low dimensional space to model the image-text relationship as well as the label redundancy. The RNN model is employed as a compact yet powerful representation of the label co-occurrence dependency in this space. It takes the embedding of the predicted label at each time step and maintains a hidden state to model the label co-occurrence information. The a priori probability of a label given the previously predicted labels can be computed according to their dot products with the sum of the image and recurrent embeddings. The probability of a prediction path can be obtained as the product of the a-prior probability of each label given the previous labels in the prediction path [8].

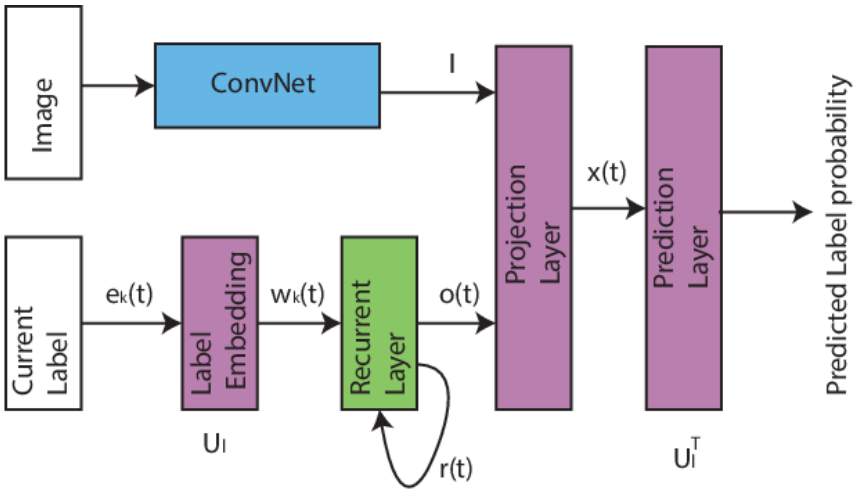


Figure 8: Overview of the CNN-RNN architecture [8].

A label k is represented as a one-hot vector $e_k = [0, ..., 0, 1, 0, ..., 0]$, which is 1 at the k-th location, and 0 elsewhere. The label embedding can be obtained by multiplying the one-hot vector with a label embedding matrix U_l. The k-th row of U_l is the label embedding of the

label k.

$$w_k = U_l \cdot e_k. \tag{8}$$

The dimension of w_k is usually much smaller than the number of labels. The recurrent layer takes the label embedding of the previously predicted label, and models the co-occurrence dependencies in its hidden *recurrent states* by learning non-linear functions:

$$o(t) = h_o(r(t-1), w_k(t)),\ r(t) = h_r(r(t-1), w_k(t)) \tag{9}$$

where $r(t)$ and $o(t)$ are the hidden states and outputs of the recurrent layer at the time step t, respectively, $w_k(t)$ is the label embedding of the t-th label in the prediction path, and $h_o(.)$, $h_r(.)$ are the non-linear RNN functions [8].

The output of the recurrent layer and the image representation are projected into the same low-dimensional space as the label embedding.

$$x_t = h(U_o^x o(t) + U_I^x I), \tag{10}$$

where U_o^x and U_I^x are the projection matrices for recurrent layer output and image representation, respectively. The number of columns of U_o^x and U_I^x are the same as the label embedding matrix U_l. I is the convolutional neural network image representation.

Finally, the label scores can be computed by multiplying the transpose of U_l and x_t to compute the distances between x_t and each label embedding.

$$s(t) = U_l^T x_t. \tag{11}$$

The predicted label probability can be computed using soft- max normalization on the scores [8].

In our use case, our CNN layer consists of three 2D CNN layers that perform the feature extraction through the whole process. With a kernel size of 5, each layer performs the convolution over the different frames, starting from the number of channels of each image (3) to 10 output channels for the first one, from 10 to 20 channels for the second and from 20

to 30 channels for the last one. Then, an average pooling layer of kernel size 4 condensates the information gathered along the image. The resulting feature vector is finally flattened to feed the coming LSTM layer.

In order to reduce time complexity in the computation and keep an appropriate convergence rate, the size of each frame is set to 35×35. It therefore follows that the input feature's size will be 750 and one would like to obtain hidden features of size 100. After flattening the LSTM's output, a fully connected layer that computes the weights for going from $100 \times Sequence\_Length$ feature dimension to the number of classes will be applied on the resulting vector, followed by a softmax layer to return the predicted class.

4.2.3 Vision Transformers

Transformers have an encoder-decoder structure. Their encoder maps an input sequence of symbol representations $(x_1, ..., x_n)$ to a sequence of continuous representations $\mathbf{z} = (z_1, .., z_n)$. Given $\mathbf{z}$, the decoder then generates an output sequence $(y_1, ..., y_m)$ of symbols one element at a time. At each step, the model is auto-regressive, consuming the previously generated symbols as additional input when generating the next. They follow this overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder, shown in left and right halves of the diagram below (see Fig. 9).

Encoder and Decoder Stacks Concerning the encoder, it is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. Applying a layer normalization between each sub-layer, their output is then LayerNorm(x + Sublayer(x)), where Sublayer(x) is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension $d_{model} = 512$. Regarding the decoder, it is also composed of a stack of $N = 6$ identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, the layer normalization between each sub-layer is employed. The self-attention sub-layer in the decoder stack is also modified to prevent positions from attending to subsequent positions. This masking, combined with fact that the

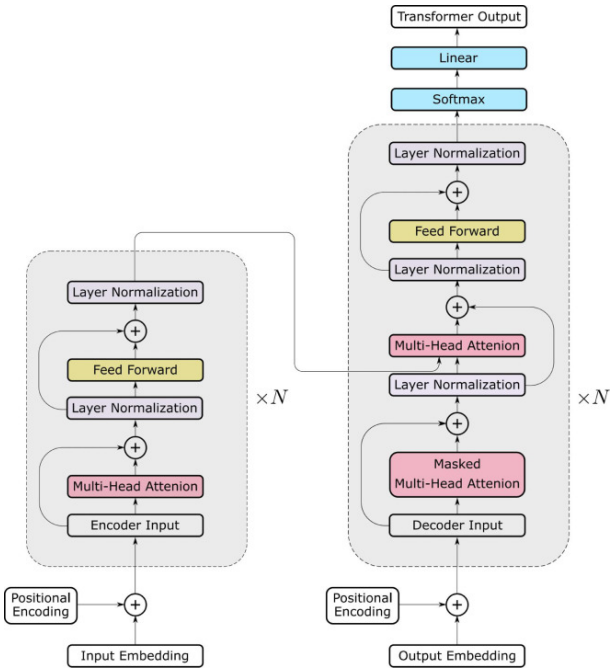


Figure 9: Model architecture of a Transformer [5].

output embeddings are offset by one position, ensures that the predictions for position i can depend only on the known outputs at positions less than i [7].

Attention An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key [7].

Scaled Dot-Product Attention This particular type of attention module needs to receive queries and keys of dimension d_k, and values of dimension d_v as an input. The principle is to compute the dot products of the query with all keys, divide each by $\sqrt{d_k}$, and apply a softmax function to obtain the weights on the values. In practice, one computes the attention function on a set of queries simultaneously, packed together into a matrix Q. The keys and values are also packed together into matrices K and V. The matrix of outputs is then given by:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{12}$$

It seems that for large values of d_k, the dot products grow large in magnitude, pushing

the softmax function into regions where it has extremely small gradients. This is the reason why the dot products are scaled by $\frac{1}{\sqrt{d_k}}$ [7].

Multi-Head Attention Instead of performing a single attention function with d_{model}-dimensional keys, values and queries, projecting linearly the queries, keys and values h times with different, learned linear projections to d_q, d_k and d_v dimensions, respectively gives more advantages. On each of these projected versions of queries, keys and values one then performs the attention function in parallel, yielding d_v-dimensional output values. These are concatenated and once again projected, resulting in the final values, as depicted in Figure 9. Multi-head attention allows the model to jointly attend to information from different representation sub-spaces at different positions. With a single attention head, averaging inhibits this. For the experimental setup, one defined $h = 8$ parallel attention layers or heads. For each of these, one used $d_k = d_v = d_{model}/h = 64$. Due to the reduced dimension of each head, the total computational cost is similar to that of single-head attention with full dimensionality [7].

Position-wise Feed-Forward Networks In addition to attention sub-layers, each of the layers in the encoder and the decoder contains a fully connected feed-forward network, which is applied to each position separately and identically. This consists of two linear transformations with a ReLU activation in between.

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \tag{13}$$

The dimensionality of input and output is $d_{model} = 512$, and the inner-layer has dimensionality $d_{ff} = 2048$ [7].

Sequence Encoding A common way to encode a sequence before passing it to a Transformer is to use positional encoding techniques to add information about the position of the processed frame in the sequence. It is also interesting to extract features of the sequence with a pre-trained CNN or even obtain temporal information of the sequence using a 3D CNN as encoding technique before feeding the Transformer.

5 Experimental Protocol

5.1 Dataset

As mentioned earlier, there are very few datasets corresponding to the needs of the project. Ideally, it could have been interesting to handle a dataset annotated for risk analysis in a railway, which does not exist yet. So the idea was to choose a commonly used dataset in Human Action Recognition, easy enough to manipulate, which means not too much videos to import and a detailed way to load sequences. KTH [6] dataset met all these expectations.

KTH dataset consists of 600 video files with a spatial resolution of 160x120 pixels, divisible in 2391 sequences (frames). It depicts altogether 6 different actions: Boxing, Hand clapping, Hand waving, Jogging, Running and Walking, which are performed by 25 different subjects in 4 different scenarios: outdoor, outdoor with scale variation, outdoor with different clothes and indoor. Another advantage with KTH dataset is the fact that it already possesses its own split for the training, validation and test sets, that is, the 11th to the 18th person (8 people) for the training set, the first, the fourth, from 19th to 21st subject and from 23rd to 25th subject (8 people) for the validation set and the second, the third, from 5th to 10th subject and the 22nd person (9 people) for the test set (see Fig. 10). There are 760 sequences for training, 768 for validation, and 863 of them for testing.

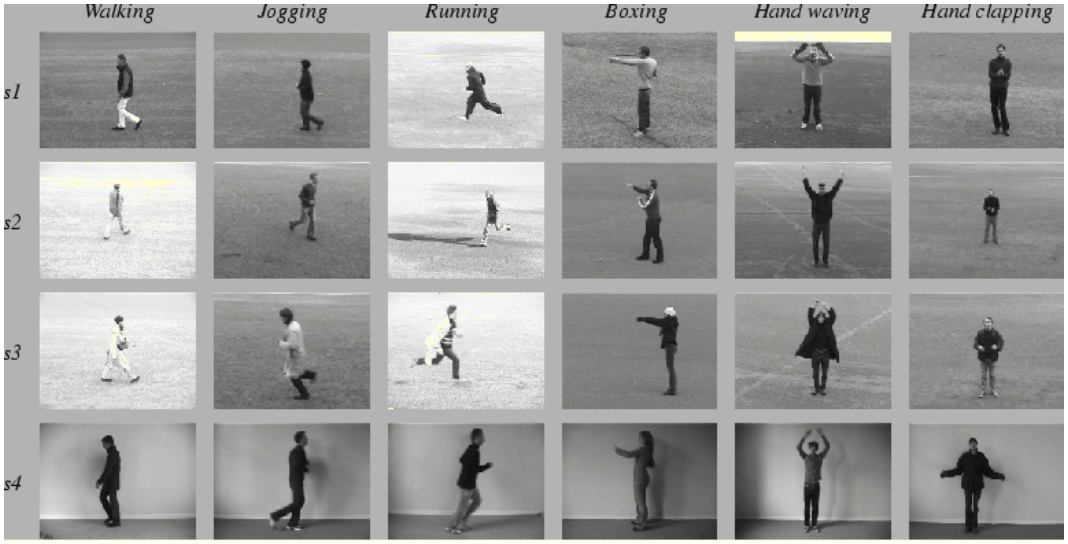


Figure 10: KTH video dataset.

5.2 Implementation Details

Performing video classification on KTH dataset, a GPU is used to facilitate the computations. Having only several hundreds of sequences to process, a batch size of 2 is sufficient to load each set of the split. Concerning the training set, consisting of 760 sequences, a shuffle operation is applied for optimizing the training conditions. Working mostly with RGB features, the number of channels will be 3. It is important to notice that in case one is working optical flow as input feature, the number of channels would be 2 (for the magnitude and the angle). The experiments are run over 100 epochs or 200 depending on the model's speed of convergence. Adam is used as the optimizer with a learning rate of 10^{-3} and a weight decay of 10^{-4}. The loss function is the Binary Cross Entropy with a label smoothing of 0.1.

5.3 Large Sequence Classification

5.3.1 3D CNN Experiments

A first idea was to gauge 3D CNN's efficiency at classifying actions behind a sequence of frames using different frame selection techniques: Left Padding, Right Padding, K-first frames selection, Motion and Attention Guided frames of interest selection. Not to forget that the ultimate goal is to reduce the relevant number of frames to the most significant ones in terms of information, so that the classification remains efficient while having fewer images to process.

In order to have a robust evaluation of the accuracy of this model, a Cross Validation has been performed by Leaving each time One Subject Out of the training set (LOSO). This experimental setup led to very interesting results, ensuring the reliability of the 3D CNN model. The padding methods, left and right, gave the same accuracy, 86.23%, with a slight advantage for the left one, whose standard deviation (std) was 6.42 against 7.33 for the right one. What concerns the k-first and motion guided methods, with a number of selected frames of 75, the previous accuracy was nearly achieved, with 84.77% (5.26 std) and 85.94% (6.36 std) respectively (see Table 1). An illustration of 3D CNN performances is showcased below (see Fig. 11).

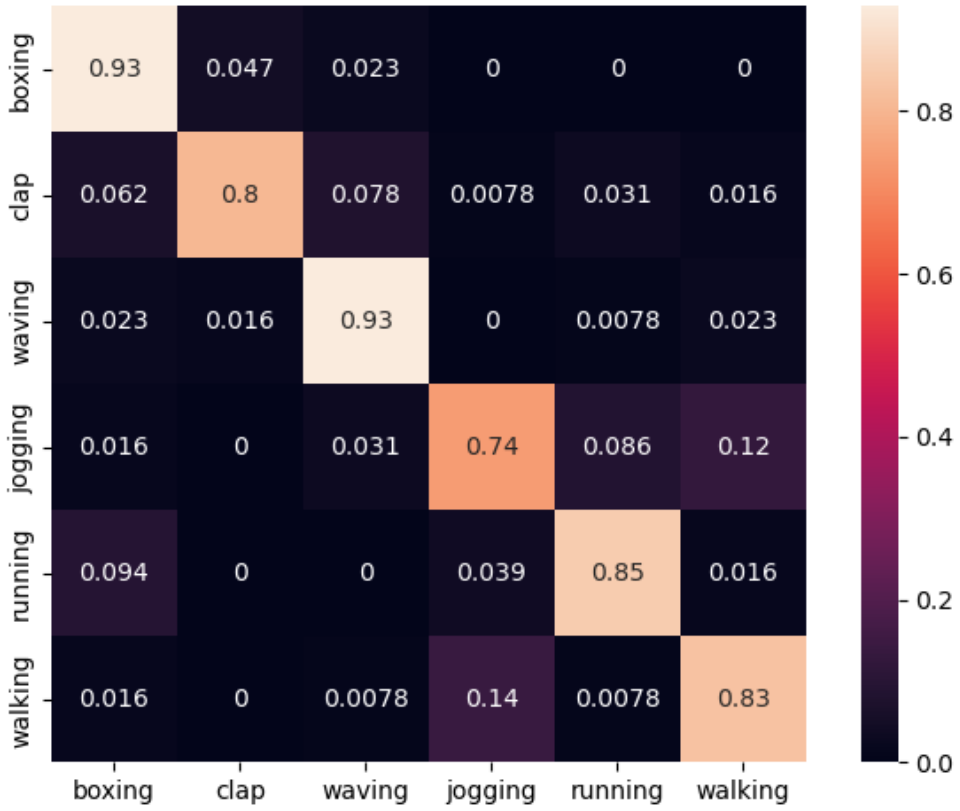


Figure 11: 3D CNN Confusion Matrix.

5.3.2 RNN Experiments

The second step was to study the behaviour of the chosen RNN model, which consists of a 2D CNN layer for encoding the sequences and a LSTM layer to classify them. The same frame selection techniques have been used to evaluate the model's robustness and the frame selection's effectiveness.

For this experimental setup, the suggested split for KTH dataset has been applied. It has shown that the model converges and led to interesting results, ensuring the reliability of the CNN/LSTM model. The padding methods, left and right, gave 73% and 70% respectively. What concerns the k-first and motion guided methods, with a number of selected frames of 100, the accuracies were roughly the same, with 70% and 74% respectively (see Table 1). An illustration of LSTM performances is showcased below (see Fig. 12).

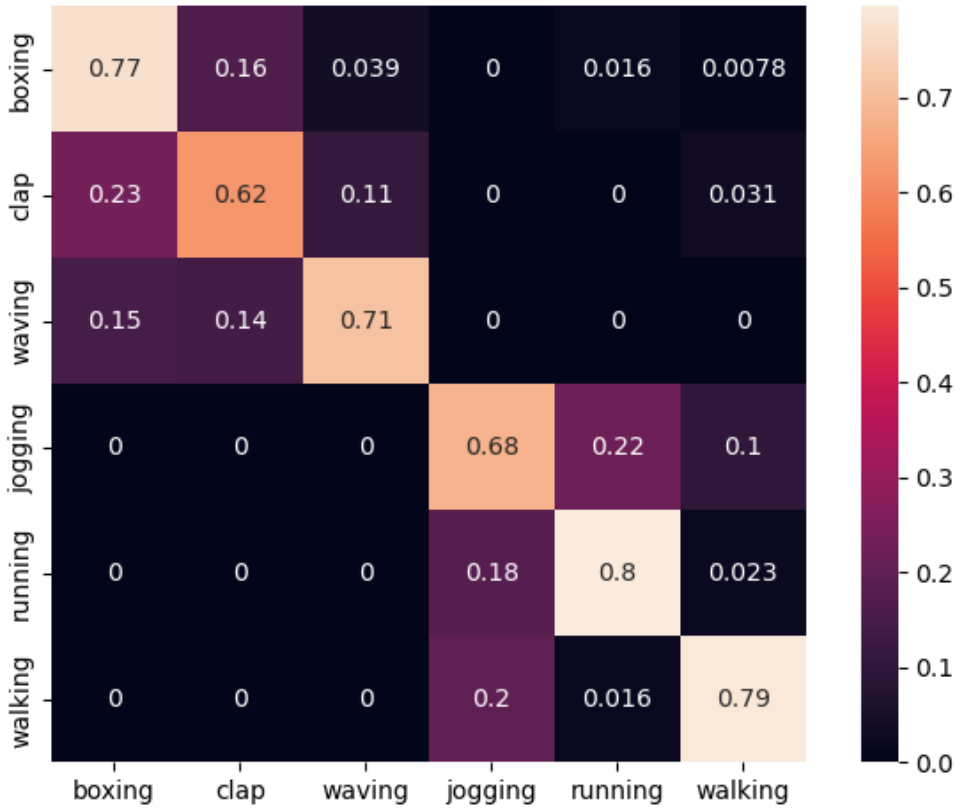


Figure 12: LSTM Confusion Matrix.

5.3.3 Vision Transformers Experiments

The third step was to observe the behaviour of the Vision Transformer (ViT) model with respect to all frame selection techniques that have been used to assess the model's robustness, the frame selection's effectiveness and the reliability of the given attention weights for better and deeper frame selection.

For this experimental setup, the suggested split for KTH dataset has also been applied. It has shown that the Transformer also converges and gave interesting results, ensuring its efficiency. The padding methods, left and right, gave 73.96% and 74.35% respectively. What concerns the k-first and motion guided methods, with a number of selected frames of 75, the accuracies were roughly the same, with 77.47% and 80.47% respectively (see Table 1). An illustration of Vision Transformer performances is showcased below (see Fig. 13).

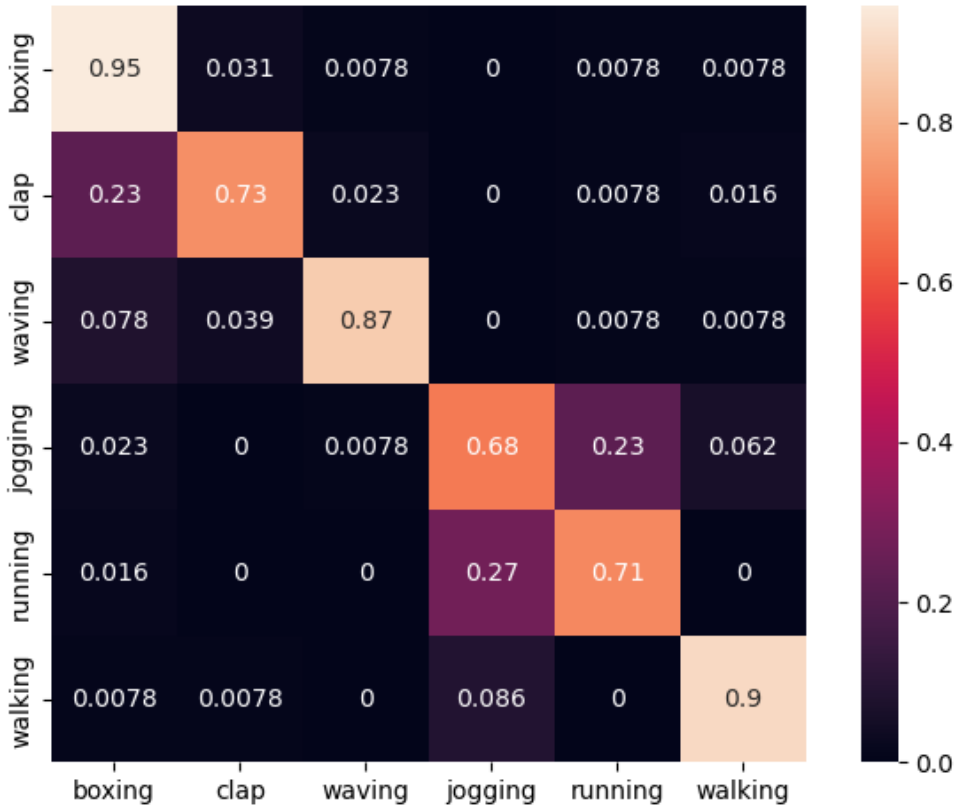


Figure 13: ViT Confusion Matrix.

5.4 Short Sequence Classification

5.4.1 3D CNN Experiments

After testing methods with relatively large number of frames per sequence, regarding the attention guided frame selection, less than 50 frames have been retained in each sequence according to what the attention module has been focusing on while browsing a sequence. The frames with the most attention weights are the frames of interest. A test has been realized with 20, 30, 40 and 50 frames successively. One could notice that with fewer frames, the 3D CNN kept performing good classification with accuracies that are not too far from those obtained with more frames per sequence: 75.94%, 78.65%, 77.47% and 80.34% respectively (see Fig. 15).

5.4.2 RNN Experiments

Having tested large number-based methods of selection, for the attention guided frame selection, less than 50 frames have been retained as well to compare with the 3D CNN. A test has also been realized with 20, 30, 40 and 50 frames successively. One could notice that with fewer frames, the RNN model also kept performing good classification with accuracies that are very close from those obtained with more frames per sequence: 69.14%, 71.22%, 72.53% and 74.22% respectively (see Fig. 15).

5.4.3 Vision Transformers Experiments

As for the attention guided frame selection, due to the tiny amount of time remaining to launch new tests on the Vision Transformer to study its capability at predicting actions with fewer frames, it could have not been possible to see the results it gives when analyzing sequences with selected frames based on their attention.

(a) K-first sequence. (b) Motion Guided sequence. (c) Attention sequence.

Figure 14: Same sequence, different frame selection technique.

Models' accuracy					
	Left Padding	Right Padding	K-first	Motion Guided	Attention Guided
3D CNN	86.23%	86.23%	84.77%	85.94%	80.34%
LSTM	73.16%	70.43%	70.12%	74.37%	74.22%
ViT	73.96%	74.35%	77.47%	80.47%	unknown

Table 1: Models' accuracy with respect to the frame selection methods.

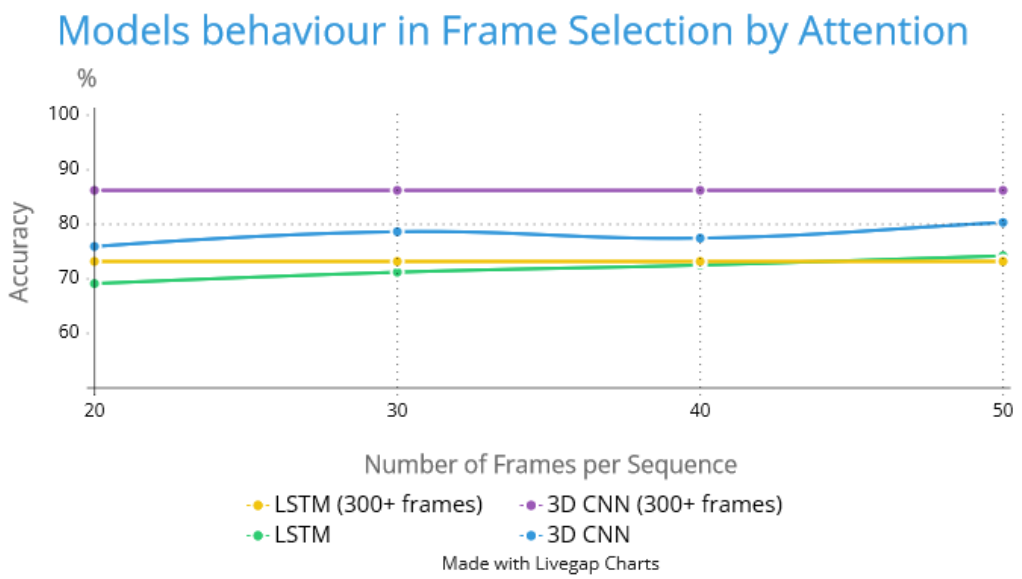


Figure 15: 3D CNN and LSTM performances with attention-kept key frames.

6 Results Discussion

Considering the obtained results, one may first notice the reliability of the state-of-the-art methods in Human Action Recognition. The three of them give results that are quite satisfying for almost all methods of frame selection that have been tested so far.

Furthermore, one may also remark on the fact that the best achieved accuracies which are the ones exploiting the longest sequence, consisting of 384 frames, to use a padding (right or left) on the remaining sequences, take benefit from large length sequences.

Still, once one uses other methods for frame selection implying less than 100 sequences, it clearly appears that the obtained result is not very far from the one obtained with full length sequences. For instance, if the LOSO protocol leads to an accuracy of 86.23% for padding methods, it also returns 84.77% for the k-first frames method, which means one can always achieve satisfying results in Action Recognition with fewer frames per sequences.

Moreover, to continue developing this idea, when it comes to Attention Guided frame selection, which consists in taking for each sequence the k frames of interest according to the attention module, one may see that for only 20, 30, 40 or 50 frames, the 3D CNN and LSTM models still give results that are closed to the ones obtained with more than 300 frames per sequence (see Fig. 15).

This shows that one does not necessarily need hundreds of frames per sequence to perform good video classification. Only a few relevant frames are required. Besides, this thematic becomes a crucial interrogation nowadays regarding the debates around climatic issues. Running training sessions with GPUs for a model processing hundreds of images per sequence is really expensive and highly contribute in the global warming in the world. Now, having a refined way to extract frames of interest to reduce the length of the sequences would not only allow to keep the same consistence in the obtained results, but also improve global gestion of climatic issues raised by the actual use of heavy computation programs.

7 Conclusion & Perspectives

In a nutshell, Frame Selection remains an open question in Artificial Intelligence and Computer Vision. Throughout the entire mission, it has been possible to explore numerous methods allowing to do so. Among them, one can distinguish handcrafted methods from Deep Learning ones. Concerning handcrafted methods, Padding sequences is the most commonly used, but picking frames of interest by means of magnitude calculation (Motion Guided) allows to reduce considerably the sequence length on one hand, and catch relevant information about the motion by having key frames on the other. Regarding Deep Learning methods, there are still lots of techniques to find out in the Literature, even though the main work of the project focused itself on a very simple and well-spread method that consists in retrieving attention weights from each sequence using a Vision Transformer that has been trained at recognizing motion and using them to select the frames whose weights are the most consistent.

To go further, now it has been proved that reducing consequently sequence length without affecting too much the results is possible, an extension of the project could be to substitute RGB frames by optical flow to have a better trace of the filtered motion and perhaps improve the performances a little more. This could be an interesting direction to give to the project, since optical flow is meant to give a real feedback of the global motion, focusing of the main parts of the image that are in displacement along the sequence.

In conclusion, this internship mission has been a real opportunity and allowed me to grow in competence and maturity. I have been able to learn a lot of topics that have not been taught in M1 and that are also coming in M2. I could have improved my own programming skills, learned about Deep Neural Networks that have not been explored during the course, such as Transformers or 3D CNNs. I have also discovered programming with PyTorch and research environment through papers and articles. There are still much improvement to do since my next objective is to organize my workspace and make it more structured to help retrieve important results. Even so, the progress that has been realized is quite satisfying and makes me feel confident about challenges coming next year.

8 Acknowledgments

A particular thought of gratitude is firstly addressed to my internship supervisors, Omar Ikne, PhD student, and Benjamin Allaert, lecturer-researcher, of the research team HIDE, who accompanied me through the whole project and guided my efforts so that the mission could be fulfilled.

My profound gratitude is also addressed to my academic supervisor, Pierre Tirilly, for his help, involvement and supervision during the internship. Without his implication, the pressure, workload and administrative issues related to internal concerns would have been far more difficult to face.

I would like to thank Justin Bescop, PhD student of the research team HIDE for his support in my work and all information he gave to me in order to grab better the idea behind the project and its applications.

I thank the whole research team of CERI SN for the very cheerful moments we spent together, the meals we shared, the games we played and the precious information we shared.

Bibliography

[1] Shivangi Anthwal and Dinesh Ganotra. "Optical Flow Estimation in Synthetic Image Sequences Using Farneback Algorithm". In: *Advances in Signal Processing and Communication*. Ed. by Banmali S. Rawat et al. Singapore: Springer Singapore, 2019, pp. 363–371. ISBN: 978-981-13-2553-3.

[2] J Arunnehru, G Chamundeeswari, and S Prasanna Bharathi. "Human action recognition using 3D convolutional neural networks with 3D motion cuboids in surveillance videos". In: *Procedia computer science* 133 (2018), pp. 471–477.

[3] Yanbo Gao et al. "A Deep Attention Model for Action Recognition from Skeleton Data". In: *Applied Sciences* 12.4 (2022). ISSN: 2076-3417. DOI: 10.3390/app12042006. URL: https://www.mdpi.com/2076-3417/12/4/2006.

[4] Shreyank N Gowda, Marcus Rohrbach, and Laura Sevilla-Lara. "Smart frame selection for action recognition". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 2. 2021, pp. 1451–1459.

[5] Kelei He et al. "Transformers in medical image analysis". In: *Intelligent Medicine* 3.1 (2023), pp. 59–78. ISSN: 2667-1026. DOI: https://doi.org/10.1016/j.imed.2022.07.002. URL: https://www.sciencedirect.com/science/article/pii/S2667102622000717.

[6] Soo Min Kang and Richard P. Wildes. *Review of Action Recognition and Detection Methods*. 2016. arXiv: 1610.06906 [cs.CV].

[7] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[8] Jiang Wang et al. "CNN-RNN: A Unified Framework for Multi-Label Image Classification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[9] Zuxuan Wu et al. "Adaframe: Adaptive frame selection for fast video recognition". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1278–1287.

[10] Yuan Zhi et al. *MGSampler: An Explainable Sampling Strategy for Video Action Recognition*. 2021. arXiv: 2104.09952 [cs.CV].