Deep Learning - based Models for Performing Multi-Instance Multi-Label Event Classification in Gameplay Footage

***Abstract*—The ability of autonomous agents to extract relevant information about the environment they operate in is essential for their good performance. In dynamic environments, like videos, one of the key pieces of information are the events, since, in a broad manner, they represent the dynamic changes and interactions that happen in the environment. Video games stand out among the most suitable domains to investigating the effectiveness of machine learning techniques. A challenging activity explored in such research consists on endowing automatic game systems with the ability of perceiving, in recorded test games, the events that other players, interacting with them, provoke in the game environment. Then, this work has as its main purpose to produce deep learning - based models to deal with multiple-instances multi-labels (*MIML*) event classification in gameplay footage. The architecture of these models is based on a data generator script, a feature extractor convolutional neural network (CNN) and a deep classifier neural network. The models are trained from datasets in which the game scenes are represented either by frames or chunks (group of frames). *Super Mario Bros* is used as a case study. The main contribution of this work are: 1) Implementation of an automatic data generator script that produces the frames from the game footage; 2) Generation of a fine-tuned MobileNetV2, from the standard MobileNetV2, specialized in dealing with gameplay footage; 3) Implementation of deep learning models which perform *MIML* event classification in gameplay footage.**

***Index Terms*—MIML event classification, gameplay footage, Super Mario, frame-based and chunk-based data representations, deep extractor Neural Networks, fine tuned backbone, deep classifier Neural Networks.**

I. Introduction

In machine learning (ML), the ability of autonomous agents to retrieve relevant information about the environment they operate in is an indispensable requisite, since it can be very useful to improving their decision making process [19]. One of the most relevant information to be retrieved from the environment in which agents in general operate refers to the ongoing *events*. Despite the complexity that the event definition can assume in Artificial Intelligence (AI) [22], [29], in short, events can be seen as *flags* that depict the most important things happening in the environment.

In areas like personal assistance robots, camera monitoring networks and autonomous cars, instantly being able to react to events happening in the environment is crucial. This is why AI researches carried out in the domain of online event detection systems can be very valuable [7].

Indeed, such challenge has motivated recent AI researches on developing online event detection systems for real world videos, be it sports, traffic, or every day activities [6], [7], [28].

For agents operating in dynamic environments, like videos, the events, which describe the dynamic changes and interactions that happen in the environment, represent fundamental information to guide their decision-making. In such situations, as the game environment varies, the agent can drastically change its decision making, which also causes a great variation in the events present in the game scenes.

Depending on the problem the studies carried out with the objective of retrieving events from videos cope with, they will perform a different kind of classification activity. Such variation will depend on the following factors: on the way the video scenes will be presented to the system, and on the number of events intended to be retrieved from such scenes.

In this way, the systems implemented in such studies can be divided into the following groups: 1) single instance, single label (*SISL*), where a scene is presented to the system through a single *frame* from which at most one label can be retrieved; 2) single instance, multi labels (*SIML*), where a scene is presented through a single *frame* from which more than one label can be retrieved; 3) multi-instance, single label (*MISL*), where a group of scenes is presented to the system through a set of *frames* from which at most one label can be retrieved; 4) multi-instance and multi-label (*MIML*), where a group of scenes is presented through a set of *frames* from which more than one label can be retrieved.

Video games have been proving to be a very appropriate case study for investigating the effectiveness of ML techniques for several reasons, among which stand out: economically, they are part of the group of products that generate the highest profits in the entertainment industry [8]; in education, they have been showing as a friendly, efficient, and attractive tool; technically, they present a very high difficulty level, which represents a great challenge to any ML approaches used to deal with them [2].

Also, a lot of games can be used to emulate practical real world situations, which correspond to an interesting alternative to deal with such situations in a controlled, easily scalable and adjustable way [16].

At last, video games have also been used in the fields of medicine and psychology as, among other things, valuable tools to study behavior, improve motor skills (especially hand-to-eye coordination), and assist with the treatment of developmental disorders [26], [12], [15], [3].

Researches aimed at producing player agents capable of recovering, from game videos, the events provoked in the game

environment by the actions executed by the opponents, can be extremely useful. Indeed, for example, in the construction of player systems aimed at improving the cognitive abilities of their users, the results of the analysis of these events can be used to map the users' profile, and this profile can be used to train the player system to adjust its decision-making process so as to lead the game to situations that stimulate the development of the users' cognitive abilities.

As argued in [2], despite the fact that, frequently, the events can be directly retrieved from the game system flags (named as game logs), the following obstacles can make such information retrieval unfeasible: firstly, game engines are frequently unavailable, due to the companies' privacy concerns [18]; secondly, in studies aiming at retrieving events from gameplay footage, which are not real-time games, it is not possible to count on game information provided by the game engines, even if the game platform they deal with count on open game engines.

The great applicability of systems capable of recovering events in video games, associated with this obstacle to accessing game logs, are factors that have been motivating new studies aimed at creating automatic systems with such ability.

In the following, some state-of-the-art works conceived to deal with retrieving events from images or videos are introduced, in virtue of being very closely related to the problem or approaches which this work deals with, which makes them essential to understand the contributions provided by the present paper.

In [17], the authors proposed a deep neural network (NN) to performing *SIML* event classification (the work did not explore *MIML* classification) in game footage of *Gwario* (a Super Mario's clone). More specifically, a single standard AlexNet was trained from scratch, with a manually labeled Gwario game dataset, with the purpose of performing both feature selection and classification from individual frames retrieved from Gwario videos.

In [5], the authors took the general *MIML* framework proposed in [31], which allows for making *MIML* classification in various domains, and used it to producing a new *Deep MIML Network* as a general Deep Learning (DL) model (trained from numerous and varied real datasets) to classifying events in images and texts (such work did not investigate event retrieval in videos).

As a precursor to the present proposal, the state-of-the-art work [2] should be introduced in a little more detail. In such work, the authors investigated 26 DL- based models to identify events in gameplay footage depicting Super Mario runs (Mario AI Framework [13]). Their main objective was to find the most adequate architecture and game scene representation to perform single event retrieval from Super Mario videos. As a preliminary study, unlike what is intended here, such work did not cope with multi label retrieval. Concerning the architectures, one of the investigated models was the single standard AlexNet-based model proposed in [17], where the AlexNet performed both the feature extraction and the classification. The remaining models combined a CNN automatic feature extractor (MobileNetV2, ResNet50V2, VGG16 or AlexNet) - to produce a concise and adequate feature-based representation for the game scenes - , and a NN game event classifier (long short-term memory (LSTM), gated recurrent unit (GRU) or fully-connected layers (FCL) - to identify the game event occurring in the feature-based representation produced by the CNN extractor. Concerning the game scene representation at the input of the CNN extractor, two kind of representation were investigated: individual frames and chunk (a set of consecutive frames). The major contribution provided by the preliminary studies in [2] was to prove that, concerning the problem of event classification in game play footage: firstly, the chunk-based representation performs better than the frame-based representation; secondly, the models which use distinct deep neural networks to perform feature extraction and event classification provide much better results than those in which a single deep neural network (CNN) carries out both activities. More specifically, the best model obtained in such work combines the *standard CNN MobileNetV2* [24], as a feature extractor NN (or backbone NN) , the *RNN LSTM* [11], as a classifier network, and *chunk* structure, as the basis for game scene representation.

Motivated by all arguments aforementioned, the present paper extends the state-of-art work [2] through the proposal of a couple of new models to perform multi label classification (instead of single label classification, as performed in the later) in gameplay footage of *Super Mario Bros* [13].

The authors' main motivation for using *Super Mario Bros* as a case study in their latest works are the results presented in [20], showing that such a game is quite suitable for the development of cognitive skills, especially for users with some type of weakness in the learning process.

By way of introducing some examples of event retrieval in *Super Mario Bros*, the occurrence of the events *EventJump* and *EventLand* indicate that *Mario has just took off* and *Mario has just landed*, respectively.

To cope with the intended objectives and taking into consideration the results obtained in the preliminary studies carried out in [2], both models proposed herein use a chunk-based representation for the game scenes, and are based on distinct deep neural networks to perform feature extraction and multi label classification. In order to extending and to improving such results, both proposed models, in addition to performing *MIML* classification, seek to enhance the best architecture obtained in it.

For this, both models count on a new *fine tuned MobileNetV2* produced here, which replaces the *standard MobileNetV2* used in [2], so as to enhance the feature extraction process.

Concerning the classifier neural network, by way of investigation, one of the models uses the LSTM, which proved to be the best single label classifier in [2], whereas the other one uses the Deep MIML, an approach proposed in [5] with the objective of dealing with multi label classification, which was tested in the same work in the domain of images and texts.

In short, the main contributions of this work are:

- Implementing two new models to perform *MIML* in game footage of *Super Mario Bros*, which extend the framework proposed in [2] by including game scenarios in which multiple events happen simultaneously;
- Implementing an automatic data generator script, which automatically generates the frames from the game footage. Such instances were used to produce adequate multi labeled instances to train the proposed models.
- To boost the feature-extraction process, this work produced a fine tuned version of the standard MobileNetV2 used in the preliminary studies carried out in [2]. Such fine tuned version was obtained by retraining the standard MobileNetV2 from a specialized dataset composed of *Super Mario Bros* frames, and improved significantly the classification accuracy, since the ImageNet dataset, from which the standard MobileNetV2 was trained, differs a lot from the visual characteristics of the Super Mario game.
- In order to investigate an alternative classifier deep NN to deal with *MIML* problem in game play footage of *Super Mario Bros*, in addition to testing the RNN LSTM (which performed well in the single label classification scenarios investigated in [2]), the present work also tested the Deep MIML neural network proposed in [5] (it is interesting to note that, in [5], the Deep MIML was just tested in the domain of images and texts, and not in the domain of videos).

The experiments proved the significant performance gain obtained in the models with the use of the fine tuned MobileNetV2 instead of the standard MobileNetV2. In addition, regarding the classifiers, they showed that the LSTM presented slightly better results than DeepMIML in terms of Mean Avery Precision and F1-Score.

II. Theoretical Foundation

This section will present a general view of the main theoretical topics related to this work.

A. Multi-Instance Multi-Label Framework

The MIML framework, proposed in [31], is defined by each data example being comprised of multiple instances, associated with multiple-labels. Traditionally in ML, each instance of data is associated with a single label, or even, each instance with multiple labels. Figure 1 illustrates such distinct frameworks. However, for problems involving complicated examples with multiple semantic meanings, using more than one instance to represent them can allow for additional inherent patterns to the data to become more clear. So, in these situations, the MIML framework can be a more natural and appropriate way to represent the data.

An important idea in multi-instance learning are sub-concepts. When considering a broad concept like "*Africa*", it can hardly be described by only one aspect. Usually a group of cultural and environmental identifiers is required. So the concept "*Africa*" could be identified from, for example, an image of a grassland environment, coupled with lions and trees. These low-level concepts that aren't necessarily enough on their own, but when combined are capable of describing complex and broad concepts, are the sub-concepts.

1) Multi-Label Evaluation Metrics: In traditional supervised learning, as mentioned in the previous subsection, a single instance is associated to a single label. This allows for an *accuracy* metric (the percentage of examples correctly classified) to often be a good enough indicator of performance [31]. However, in multi-label situations, the goal is not to identify a single label, but rather to correctly classify the highest amount possible from a group of labels. Then, for example, it is better for a model to identify 4 out of 5 labels correctly, than getting 2 out of the same 5 labels. This has to be taken into account by a multi-label evaluation metric for it to be an effective performance indicator.

The evaluation metrics that were relevant to this work are explained below:

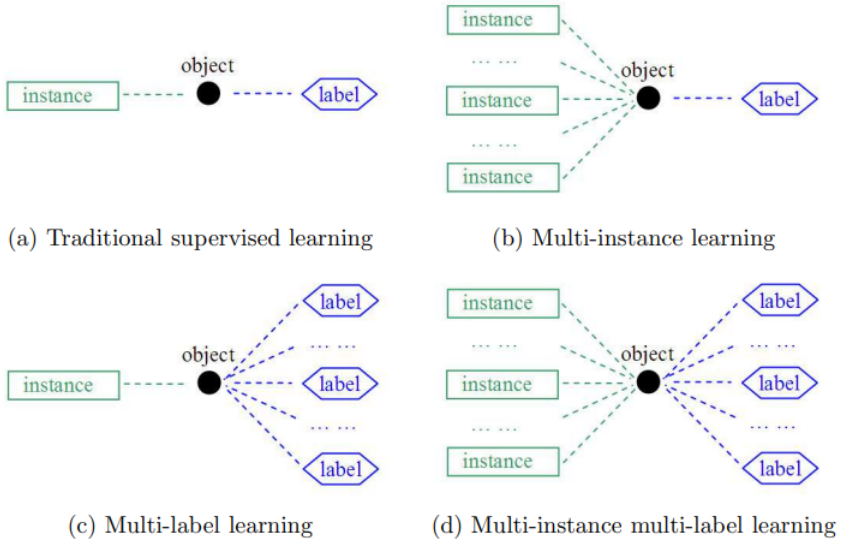
- Hamming loss: a loss metric that represents the fraction of labels that are incorrectly predicted, be it a correct label that is missed, or a wrong label that is predicted. Considering the function $hamming_loss(h) = 0$, the lower the value of $hamming_loss(h)$ the better the performance of h.
- Mean average precision: the average precision (AP) is a relation between the class precision and class recall. Precision (P) indicates how many predictions were correct and recall (R) indicates how many of all instances of a class were predicted by a model. The average precision corresponds to the area under the graph $precisionXrecall$. Since the average precision corresponds to a single class, the mean average precision (mAP) corresponds to the average APs over all classes.
- F1 score: the F1 score, also is a relation between class precision and class recall. However, it corresponds to the harmonic mean between the two, so: $F1 = 2 * \frac{P*R}{P+R}$. Like mAP, the F1 score also references a single class, so in a multi-label context, this metric corresponds to the average F1 score of all classes.

B. Deep Neural Networks

This section will provide a brief overview of the deep neural network architectures that were used in this work.

1) Convolutional Neural Networks: Convolutional neural networks (CNNs) are deep learning models specialized in image classification problems. They are composed of alternated convolutional, and pooling layers that perform an automatic feature selection process in order to transform the network's input into the best possible representation for classification performance [1]. Some of the more used CNNs are: MobileNet, ResNet and VGG-16.

2) Recurrent Neural Networks: RNNs are architectures that incorporate specific connections between neurons in order to allow for relevant information to persist in the network across multiple iterations. They are particularly effective in scenarios where the class of an instance has a correlation with the classes of previously processed examples in the network [4]. Problems like action detection in videos, multiple object detection and



(a) Traditional supervised learning (b) Multi-instance learning

(c) Multi-label learning (d) Multi-instance multi-label learning

Fig. 1. Four different learning frameworks, image from [31]

text analysis greatly benefit from this feature. Some of the more used RNNs are the long-short therm memory (LSTM) and the gated recurrent unit.

3) Deep MIML Network: As previously mentioned in section II-A, a lot of real world applications greatly benefit from being represented in a multi-instance multi-label context. The Deep MIML Network [5] is a model that adds an automatic instance generator and a sub-concept learning structure to the traditional neural network formation. The structure of the Deep MIML Network (Figure 2) consists of a feature-based-instance generator module (usually a CNN network), followed by a *Sub-Concept Layer*, which is essentially a classifier that matches the scores between instances and sub-concepts for every label. This approach allows for an instance-label relation discovery that works very well in a MIML framework.

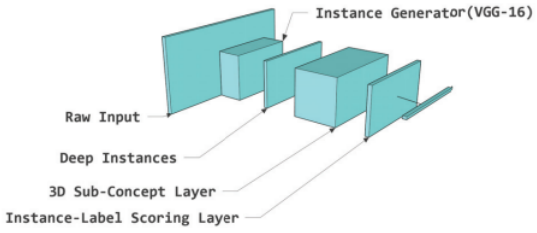


Fig. 2. Representation of a Deep MIML Network, image from [5]

C. Super Mario Bros

The game *Super Mario Bros* was first released in 1985 by the company Nintendo. In it, the player plays as a plumber called Mario and has to traverse a series of stages in order to reach and save a princess. Even though it is a very simple premise, each stage presents a different set of obstacles and enemies that require precise actions from the player in order to advance. These actions are: walking, running, jumping and throwing a fire ball. All of these trigger different *Game Events*, that represent the interactions the player has with the environment through his actions. All the game events related to Super Mario are listed in section IV-A.

From a visual standpoint, the game has very simple graphics in a ”pixelated” style. The game also presents a 2D (two-dimensional) point of view, which lacks the object depth that is present in 3D games [23] as can be seen in Figure 3.

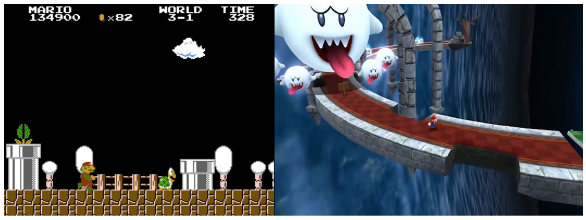


Fig. 3. *Super Mario Bros* on left (2D), Super Mario Galaxy on right (3D)

III. Related Works

This section presents a brief overview of some state-of-the-art works in the domains of *information retrieval in Video Games* and *Multi-Instance Multi-Label classification*, which dealt with problems or methods close to those the present paper copes with. It is interesting to remember that some of them, in virtue of being directly related to the present proposal, have already been introduced in Section I of this paper.

A. Information Retrieval in Video Games domain

In the work [16], the authors developed a system to generate a vehicle collision dataset from the game *Grand Theft Auto*

V, that was used to train a CNN model to detect collisions in real-world settings. In the context of information retrieval, in [27], the authors used the *Super Mario Bros* game to analyze how much data quantity and data quality impact on distinct machine learning-based methods which perform content generation. Still concerning *Super Mario Bros*, [10] proposed a new approach to learn gameplay rules, as well as game level design characteristics, from gameplay footage. The paper [17], already mentioned in Section I, proposed DL based approaches to deal with multi-label game event classification in footage of the following games: *Gwario* (a Super Mario's clone), *Megamen*, and *The Elder Scrolls V: Skyrim*. In such approaches, a single CNN (AlexNet) was used for both feature extraction and classification. Such work just investigated the frame-based representations for the game scenes (differently from the present work, it did not test the chunk-based representation or architectures composed of distinct neural networks to drive feature selection and classification). Their primary contribution was showing how CNN networks pre-trained on real-world data can have great results when applied to the correct game scenario. The work [9] had as its purpose to predicting the set of actions that an automatic player has to execute so as to reach its intended goal. For this, the plan recognition framework receives as its input a set of events representing the intended objective - provided by the game logs - and outputs the sequence of actions that must be performed in order to reach such objective. The article uses a LSTM model for dealing with both single and multi label architectures. The game used as a case study was *Crystal Island*. Differently from the present proposal, in [9] the authors count on the events provided by the game logs.

Finally, the state-of-the-art work [2], as the predecessor to this work, has also been introduced earlier, in more detail, in Section I. In short, such work investigates various deep learning-based models to identify single game events occurring in *Super Mario Bros* gameplay footage (it did not investigate multi label classification). Its main contribution was to demonstrate the greater performance reached by the models which combines the chunk representation for the game scenes with the resources of the classifier recurrent neural networks (RNN). In fact, the best model obtained combined *standard MobileNetV2+LSTM+chunk* (such work did not investigate the performance of the Deep MIML, a neural network conceived to deal with multi label classification).

B. Multi-Instance Multi-Label Classification in other Domains

In [31], the authors proposed the Multi-Instance Multi-Label framework, that is widely used to structure classification problems across many research fields. [5] took this framework and proposed a new *Deep MIML Network* as a general DL model that, not only is able to automatically generate multiple instances from a single piece of data thought a CNN, but also can keep track of instance-label relations. Such neural network was validated through experiments involving images and texts (differently from the present paper, such work did not test the DeepMIML operating in video scenarios). Also, in [25] the authors explored the usage of CNNs on multi-label image classification problems. They proposed a deep Multi-Modal CNN, that was designed for MIML learning, and whose main appeal was combining the benefits of the MIML framework, with the image classification capabilities of CNNs. In order to solve the problem of weakly labeled images in classical datasets (like ImageNet), in which images presenting multiple object classes are labeled with just a single label, In [21], the authors proposed a multi-label iterated learning to incorporate the inductive biases. This process works in two phases. In the first one, a 'teacher' model interacts with the single-label data to improve the multi-label classification (the interactions in this phase are limited in order to prevent the binary classification overfitting). The second phase uses the knowledge acquired on the first one to train a 'student' multi-label model. The final system receives an input image and outputs the set of labels occurring in the received image. Such work did not operate with multi label classification in videos.

IV. Architecture of the Models

This section details the models investigated in this work to tackle *MIML* event classification in game-play footage of *Super Mario Bros*. These proposed approaches are based on the association of a CNN backbone and a deep classifier NN. In the first stage, the aim is to automatically extract the most relevant features to represent the game scenes, whereas, in the second stage, the classifier NN is employed to identify the events in such scenes.

The approaches were created so as to extend the results obtained in the following state-of-the-art works: 1) In [2], the authors investigated various models to perform single event detection in game-play footage of *Super Mario Bros* (differently from the present paper, such work did not cope with multi label classification). These investigated models differed from each other according to the following aspects: first, in the way of representing the game scenes: frames or chunks; secondly, in the composition of the architecture of the models: one of the architectures was composed of a single deep network to perform both feature extraction and single event classification, and a second alternative architecture composed of two distinct networks, where a CNN performed feature extraction and another network (a deep RNN or a Multi Layer NN) performed the single label classification. The winner model was made of a MobileNetV2 feature extractor combined with a classifier LSTM, and used chunks to represent the game scenes; 2) In [5], the authors proposed the DeepMIML 2D Sub-concept Layer to perform multi-label classification in images and texts (distinctly from this work, they did not investigate the performance of DeepMIML executing *MIML* classification in videos).

Then, in order to cope with the objective of performing *MIML* classification in game footage of *Super Mario Bros*, and taking into consideration the results obtained in such state-of-the-art-works, both models proposed in this work, additionally to use chunks to represent the game scenes, count on two distinct NN to perform feature extraction and multi label

classification. The architecture of these models are resumed in the following:

- Both models count on an automatic data generator (script): the script has as its purpose to generate the data that will be used to build the training datasets.
- CNN backbone: in both models, a new *fine tuned MobileNetV2* was implemented with the purpose of replacing and enhancing the *standard MobileNetV2* backbone used in [2]. Such a new version was produced by retraining the *standard MobileNetV2* from a dataset composed of the frames produced by the automatic data generator (script);
- Multi-label classifiers: by way of investigation, one of the models uses the LSTM, which proved to be the best single-label classifier in [2], whereas the other one uses the Deep MIML proposed in [5], in such a way as to evaluate to which extent it performs well in making *MIML* in videos.

The details of such *MIML* classification from game play footage will be explained in detail in the following subsections.

A. A Script to Generate the Training Datasets

As this work copes with a multi-label classification problem, it has to count on a significant quantity of multi-labeled samples to train the deep classifier NNs (Section IV-C2). Then, it was necessary to implement a script that automatically retrieved data from gameplay footage. For this, the script, through adequate resources provided by the Mario AI Framework [13], produced the gameplay footage from which the following elements were extracted: the data corresponding to frames which represent the game scenes, and some CVS files containing the labels associated with these frames.

The footage is composed of 75 videos, each one containing the recording of the same group of 15 levels of *Super Mario Bros* played by a different competitor picked up from a set of 5 players, among which 4 are humans and one is the automatic player available in the Mario AI Framework.

The script had also to define both the *framerate* of the footage - which represents how many times the game scene refreshes per second of the game - and the set of events that is adequate to label the data samples.

Concerning the *framerate*, the footage was captured at 30 frames per second. It means that the time interval between two consecutive frames is about just 0.03 seconds, which very frequently makes the script generate frames devoid from events (since there is not time enough to allow the occurrence of an event in consecutive frames).

In order to label the frames, the script used the same 12 game events pointed out by the Framework itself as being fundamental to model the dynamics of a match (such events stand out among the 270 variables generated by the Mario AI Framework to describe a given data scene, which include game events, coordinates, and others) [2].

It is interesting to note that the frames generated by the script can present zero or more events.

The game situations corresponding to these 12 game events are briefly described next:

- EventBump: Mario bumps his head on a block after jumping;
- EventCollect: Mario collects a coin;
- EventFallKill: an enemy dies by falling out of the scene;
- EventFireKill: Mario kills an enemy by shooting fire at it;
- EventHurt: Mario takes damage after being hit by an enemy;
- EventJump: Mario performs a jump;
- EventKick: Mario kicks a Koopa shell;
- EventLand: Mario lands on the ground after having jumped;
- EventLose: Mario loses the game level, which can be caused by the player dying or the time running out;
- EventShellKill: Mario kills an enemy by kicking a Koopa shell at it;
- EventStompKill: Mario kills an enemy by jumping on top of it;
- EventWin: The player completes the current level (see Figure 4).

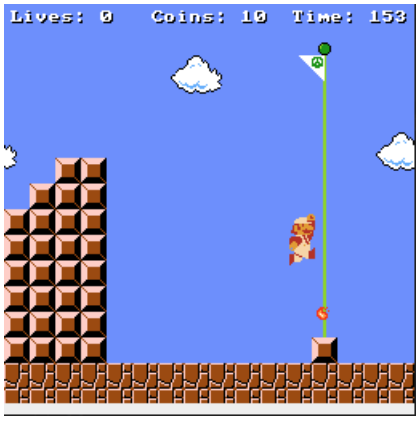


Fig. 4. Example of an *EventWin* frame.

B. Data Pre-processing

Initially, the pre-processing was performed in order to refine the row samples containing *at least one event* produced by the script, making them more suitable to enhance the quality of the training of the proposed models. In short, the following pre-processing stages were implemented:

- *Definition of an adequate capture window*: The Super Mario AI Framework registers an event that occurs in a given frame at the exact moment it is fired off in the such frame. However, for some kinds of events, such an annotation is not adequate, since, visually, the effect of triggering that event will only be noticed in a later frame. Due to this, the first pre-processing stage has as its purpose to define an adequate custom capture window for each kind of event label. For example, specifically for the event *EventBump*, instead of annotating its occurrence

in the frame f where it was fired off (as the Framework does), here, through this pre-processing stage, it will be registered in the frame that succeeds f.

- *Frame resizing*: the proposed models were implemented in the Keras API. Then, in order to make the dimension of the frames suitable for the deep NNs that were used, in this second pre-processing step, each frame was resized from its original size to 224x224 pixels, and a pixel-wise normalization was applied. All three RGB color channels were kept.

C. Training Datasets

In order to train the models proposed in this work, two datasets had to be created: a frame-based dataset and a chunk-based dataset.

For this, firstly, once the process of generating training samples has been completed by the script (Sub-section IV-A), all labeled row frames with *at least one* event that were generated were pre-processed (Section IV-B) and stored in a primary dataset. Next, as such a dataset was originally very unbalanced due to the fact that some events naturally occur much more than others in the footage from which the data were retrieved, it had to be submitted to an adequate balancing strategy. As, due to intrinsic characteristics of the *Super Mario Bros* game, about 98% of the frames with *at least one event* are single-labeled instances (that is, they present just *one* event), it was possible to use a simple oversampling balancing in which the frames belonging to the under classes were randomly duplicated until a more even distribution was reached. At the end, this pre-processed and balanced primary dataset was composed of $10,000$ examples, among which only 218 were multi-labeled frames (that is, frames with more than one event). As presented in the following subsections, both the frame-based and chunk-based training datasets were created from this pre-processed and balanced primary dataset.

1) Frame Dataset: In order to obtain a more effective feature extractor method, here the standard MobileNetV2 pre-trained from ImageNet [24] had to be retrained. To speed up such retraining, it was performed from a *Frame Dataset* exclusively made up of the $9,782$ single labeled frames present in the pre-processed and balanced primary dataset aforementioned. The idea, in this case, is to improve the backbone NN by retraining it from images specifically corresponding to the problem it will cope with, that is, *Super Mario Bros* images.

2) Chunk Dataset: The *Chunk Dataset* is used to train both multi-label classifiers investigated in this work. It is made up of multi-labeled samples (chunks) which were produced from the frames generated by the script.

Each chunk C_f of the *Chunk Dataset* was generated through the application of the following algorithm to one of the $10,000$ frames f, with at least one event, which makes up the pre-processed and balanced primary dataset (which means that $10,000$ chunks were generated):

1) Take a frame f from the pre-processed and balanced primary dataset;
2) Find f in the sequence of row frames produced by the generator script; in the same sequence, take the three frames that precede f and the three frames that follow it (it is interesting to note that such frames can be devoid of event);
3) Apply the same pre-processing described in IV-B to all the frames taken in the previous step that presents no event (which had not yet been pre-processed since they do not belong to the primary pre-processed and balanced dataset). Note that all the remaining row frames taken in the previous step (that is, those with at least one event) have already been pre-processed since they belong to the primary pre-processed and balanced dataset;
4) Build the chunk C_f by grouping, in the same order they appear in the sequence produced by the generator script, the three frames that precede f, f itself and the three frames that follow f (and all these 7 frames must already be in the pre-processed form). It is interesting to point out that such a strategy of generating chunk-based data by grouping consecutive frames retrieved from the footage by the script is very useful for two reasons: first, this produces a kind of synthesis of a temporal fraction of the game scenes, which will contribute to improving the perception of the game by the classifier deep NN and, consequently, their accuracy; secondly, it allows for significantly increase the number of multi labeled instances in the Chunk Dataset (as detailed in the following), which is very useful to enhance the multi-label classifier training.
5) Label the chunk C_f with the set of labels of its frames.

Finally, this Chunk Dataset generator algorithm presented above proved to be quite effective, since, despite the fact that a very large part of the frames with which it operated was devoid of events (Section IV-A) or, then, had just a single event (as discussed at the beginning of this section), the algorithm managed to produce a very expressive quantity of multi-labeled samples to train the classifiers. In fact, among the $10,000$ chunks that make up the *Chunk Dataset*, $3,791$ correspond to multi-labeled examples (with two or more labels), thus enabling the training of the multi label classifiers (which would have been impossible to do with the pre-processed and balanced primary dataset, which just counts on 218 multi-labeled samples, as explained in the beginning of this section).

D. Feature Extraction

Concerning the feature extraction stage, as aforementioned, the main objective here is to produce a fine-tuned version of the standard MobileNetV2 pre-trained on the ImageNet dataset [24]. The following reasons motivated to select the MobileNetV2 model as the feature extraction method: i) First, it presents a very compact architecture compared to other existing feature extractors, both in terms of a number of layers and the dimension of the weight vector, which allows for faster processing of the input data; ii) second, this model has proven its effectiveness in performing feature extraction in the winning model obtained in the preliminary studies carried out

in [2]) (which performs single event classification in *Super Mario Bros* game-play footage).

In this way, here this standard pre-trained MobileNetV2 was retrained from the *Frame Dataset* presented in Subsection IV-C. Such a strategy is based on the following fact: the ImageNet dataset, from which the standard MobileNetV2 was trained, is composed of real-world images that differ a lot from the visual aspects of Super Mario's game scenes. Then, the retraining of MobileNetV2 from the *Frame Dataset*, whose instances correspond to frames reporting the real visual aspects of the game scenes, will allow for producing a fine-tuned feature extractor able to improve the performance of the proposed models.

Based on the best model obtained in [2], here the training of the feature extractor and the deep NN classifier were performed separately, as follows: Initially, it was performed the training that produced the fine-tuned MobileNetV2; next, in order to train the multi-label game event deep NN classifier, each chunk selected in the *Chunk Dataset* was firstly submitted to the fine-tuned MobileNetV2, and the feature-based representation produced by this extractor, for that chunk, was then presented at the input of the deep NN classifier.

For this, the architecture of the fine-tuned MobileNetV2 was shortened as follows: its fully connected layer, responsible for classification, was removed (in such a way as to keep only the extractor layers intact in the architecture). This way, such a shortened architecture will be able to produce, at its extractor output layer, the feature-based representation of each chunk that must be presented at the input layer of the deep classifier NN that must be trained.

E. Classifiers

In order to find an adequate algorithm to perform classification in game-play footage, this work investigated two distinct deep NN classifiers: the sub-concept layer of the DeepMIML [5] and the LSTM [11].

The reasons for such a choice are explained in the following.

Concerning the sub-concept layer of the DeepMIMIL, as resumed in Subsection II-B3, it corresponds to the classifier portion of the DeepMIML NN architecture, which was conceived to operate with MIML problems [5]. The effectiveness of such a classifier layer to automatically generate the sub-concepts that will help to identify each label makes it quite suitable for dealing with MIML problems. Further, the fact that the present work uses multiple frames with multiple labels (chunks) as the basis for representing the game scenes, allows for including the problem it deals with in the class of the MIML problems [31], which naturally justifies the proposition of the hypothesis that the DeepMIML can be an interesting approach to be included among the classifier deep NNs to be investigated herein.

With respect to the LSTM model, it was selected for the following reasons: firstly, because it proved to be very effective as a classifier NN in the winner model obtained in the preliminary studies related to single event classification in *Super Mario Bros* game-play footage carried out in [2]. Such a good result is due to the fact that, in LSTM, the intermediary neurons are replaced with memory cells that allow storing the persistence of relevant information in the NN across the processing of different instances. Secondly, in the work that proposed the Deep MIML [5], by way of evaluating it, the authors compared its performance with that of LSTM, using as a case study a multi-label image classification problem (differently from this work, they did not investigate the performance of LSTM operating as a *MIML* classifier in videos). By a small margin, the LSTM performed better than the Deep-MIML. Finally, the LSTM is one of the most successful RNNs used in the literature until today.

V. Experiments and Results

This section presents the experiments carried out with the objectives of producing the fine-tuned backbone and the classifier NNs used in the proposed models and evaluating the performance of these models.

The performance evaluation of the models in this work will be done through the following two experiments: the first one aims to evaluate how much better the fine-tuned backbone proposed here operates than the standard MobileNetV2. The second evaluative experiment has as its purpose to compare the performance of both classifiers investigated in this paper, that is, LSTM and DeepMIML.

A. Creation and Evaluation of the Feature-Extractor

This subsection has as its purpose to present the creation of the shortened and fine-tuned MobileNetV2 conceived with the purpose of operating as a feature extractor in the models proposed in this paper (Section IV-D), as well as to evaluate to which extent it performs better than the standard pre-trained MobileNetV2.

The fine-tuned NN extractor was created and evaluated according to the following steps: 1) Firstly, a complete architecture (feature extractor layers + classifier layers) of this fine-tuned extractor NN was built from the re-training, on the *Frame Dataset* presented in Section IV-C1, of the complete architecture (feature extractor layers + classifier layers) of the standard MobileNetV2 pre-trained on the ImageNet dataset; 2) In order to make a comparison between the performance of the fine-tuned MobileNetV2 obtained in *step 1* and the standard MobileNetV2, both were submitted to a validation test from *Frame DataSet*; 3) Finally, the classifier layers of the fine-tuned MobileNetV2 obtained in *step 1* were removed, in such a way as to keep only the extractor layers, which correspond to the shortened fine-tuned MobileNetV2 to be used as feature extractor in the models proposed herein. It is interesting to point out that this shortened and fine-tuned MobileNetV2 has the same number of features (at the output extract layer) and parameters as the extractor part of the standard pre-trained MobileNetV2 [24], that is, 1,280 features and 2,257,984 parameters, as shown in Table I.

Further, the NNs used in *step1* and *step2* were implemented on Keras, and the fine-tuning training as well as the validation were performed with a 5-fold cross-validation method for up

to 100 epochs, or until there was no improvement on the loss for five consecutive epochs. The optimizer used was Adam [14], the loss parameter was categorical cross-entropy, and the learning rate was equal to 0.0005, following what was used in [2].

TABLE I
FEATURE-EXTRACTOR MODELS OVERVIEW

	Pre-Trained Weights	Training Dataset	Output Size
MobileNetV2	ImageNet	-	1280
MobileNetV2-FineTuned	ImageNet	Frame Dataset	1280

The comparison between both feature extractor NNs was made through the same evaluative parameters normally used in the state-of-the-art [2] to compare the performance among backbone NNs, that is, the *Mean Accuracy* and the *Mean Loss*. Table II shows the results obtained in *step 2*, which allows us to observe that the fine-tuned MobileNetV2 model proposed in this work (accuracy 90.81% and Mean Loss 0.283%) performs better than the standard MobileNetV2 model (accuracy 74.69% and mean loss 0.702%). This makes sense, considering that ImageNet, from which the standard MobileNetV2 model was trained, is a dataset comprised of real-world images, whereas the *Frame Dataset* used to train the fine-tuned MobileNetV2 model was composed of images related to the problem faced here. Then, fine-tuning definitely proved to be a good approach to providing a feature representation more suited to the desired game aesthetic.

TABLE II
MEAN ACCURACY AND LOSS RESULTS

	Mean Accuracy	Mean Loss
MobileNetV2	74.69%	0.702
MobileNetV2-FineTuned	90.81%	0.283

Taking into consideration these favorable results, the shortened and fine-tuned MobileNetV2, from this point on, referred just as *fine-tunned MobileNetV2* model, was chosen as backbone NN in both proposed models.

B. Analysis of the Classifiers

As explained before, this work proposes two distinct models to perform MIML event classification in game footage: 1) a first one, combining the fine-tunned MobileNetV2 model as a feature extractor and the sub-concept layer of the DeepMIML as a classifier, named *Fine-tunned-MobileNetV2 + DeepMIML-SubConcept-Layer*; and 2) a second model, combining the fine-tunned MobileNetV2 as a feature extractor and LSTM as a classifier, named *Fine-tunned-MobileNetV2 + LSTM*. Then, concerning the first model, it is interesting to point out that the fine-tuned MobileNetV2 model plays the role of the feature-based-instance generator module mentioned in Section II-B3. This way, in both models, the fine-tuned MobileNetV2 provides the feature-based representation for every chunk from the *Chunk Dataset* that will be used to train the classifiers, as mentioned in Section IV-D.

In Table III are presented the results related to both classifiers, where the 12 labels mentioned in such a table refer to the 12 events described in Section IV-A. Again, both networks were implemented on Keras, and a 5-fold cross-Validation method, with a 0.0005 learning rate was used for training and testing. The chosen optimizer was *dadelta* [30].

In this second experiment, the analysis was also carried out based on the same parameters that have been used in the state-of-the-art [5] in order to compare the performance of *MIML* classifiers that is: the *Mean Average Precision*, the *F1-Score* and the *Hamming Loss* (Section II-A1).

The results obtained in the experiments are presented in Table V-B.

TABLE III
CLASSIFIERS MODELS OVERVIEW

	Pre-Trained Weights	Training Dataset	Labels
DeepMIML Sub-Concept Layer	-	Chunk Dataset	12
LSTM	-	Chunk Dataset	12

TABLE IV
CLASSIFIER RESULTS

	mAP	HammingLoss	F1
DeepMIML Sub-Concept Layer	87.92%	0.014	0.91
LSTM	89.33%	0.011	0.93

The results show that the LSTM classifier performs a little better than MIML in all evaluated metrics. In fact, the *Mean Average Precision*, the *Hamming Loss* and the *F1-Score* produced by the former and the latter were, respectively: 89.33%, 0.011%, 0.93, and 87.92%, 0.014%, 0.91. It is interesting to note that the same slight performance superiority of the LSTM model over the DeepMIML model observed here, dealing with *MIML* classification in game footage, was observed in [5], where the authors compared the performance of models based on LSTM and on MIML classifiers coping with problems related to making *MIML* classification in images.

VI. CONCLUSIONS

In order to extend the current state of the art in the domain of performing *MIML* event detection in gameplay footage (using *Super Mario Bros* game as a case study), this work proposed two distinct deep learning models, in which the instances are represented by chunks and the feature extraction is made by a fine-tuned MobileNetV2 model produced in this study. With respect to the deep NN classifier, the first model uses the sub-concept layer of the DeepMIML - so far, only tested in the domain of images and texts [5] - , whereas the second one uses the LSTM model - only investigated in [2] as a *MISL* approach to perform single label classification in footage of Super Mario. The results proved that both models succeeded in the task they dealt with, even though the *Fine-tunned-MobileNetV2 + LSTM* model

performed a little better than the *Fine-tunned-MobileNetV2 + DeepMIML-SubConcept-Layer* model. Another contribution of this work is to implement a script that automatically generates the frames that will be used to compose the training datasets (a pre-processed and balanced frame-based dataset to train the backbone fine-tuned MobileNetV2, as well as a chunk-based dataset composed of multi-labeled samples to train the classifiers).

In future works, the authors intend to use the models produced in this work in the implementation of a player agent endowed with the ability to stimulate the cognitive and/or motor skills of patients with some kind of syndrome (such as Down syndrome), in the following way: the events identified in the gameplay footage will represent the basis to perceive the actions performed by the game users and, through them, to map their possible cognitive weaknesses. From this information, the agent must adapt its decision-making engine in order to lead the game into situations that stimulate the cognitive development of its users.

REFERENCES

[1] Neena Aloysius and Geetha Ganesh. A review on deep convolutional neural networks. pages 0588–0592, 04 2017.

[2] Anonymous. Details omitted for double-blind reviewing.

[3] Elizabeth Boyle, Thomas M. Connolly, and Thomas Hainey. The role of psychology in understanding the impact of computer games. *Entertainment Computing*, 2(2):69–74, 2011. Serious Games Development and Applications.

[4] Wei Fang, Yupeng Chen, and Qiongying Xue. Survey on research of rnn-based spatio-temporal sequence prediction algorithms. *Journal on Big Data*, 3(3):97, 2021.

[5] Ji Feng and Zhi-Hua Zhou. Deep miml network. In *AAAI*, 2017.

[6] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. RED: reinforced encoder-decoder networks for action anticipation. *CoRR*, abs/1707.04818, 2017.

[7] Roeland De Geest, Efstratios Gavves, Amir Ghodrati, Zhenyang Li, Cees Snoek, and Tinne Tuytelaars. Online action detection. volume abs/1604.06506, 2016.

[8] Global Data. Video games market set to become a 300bn-plus industry by 2025. https://www.globaldata.com/video-games-market-set-to-become-a-300bn-plus-industry-by-2025, 2021.

[9] Alex Goslen, Dan Carpenter, Jonathan Rowe, Roger Azevedo, and James Lester. Robust player plan recognition in digital games with multi-task multi-label learning. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 18(1):105–112, Oct. 2022.

[10] Matthew Guzdial, Boyang Li, and Mark Riedl. Game engine learning from video. pages 3707–3713, 08 2017.

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[12] V. Janarthanan. Serious video games: Games for education and health. In *2012 Ninth International Conference on Information Technology - New Generations*, 2012.

[13] Sergey Karakovskiy and Julian Togelius. The mario ai benchmark and competitions. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):55–67, 2012.

[14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[15] Michail D. Kozlov and Mark K. Johansen. Real behavior in virtual environments: Psychology experiments in a simple virtual-reality paradigm using video games. *Cyberpsychology, Behavior, and Social Networking*, 2010.

[16] Kangwook Lee, Hoon Kim, and Changho Suh. Crash to not crash: Playing video games to predict vehicle collisions. In *ICML 2017*, 2017.

[17] Zijin Luo, Matthew Guzdial, Nicholas Liao, and Mark Riedl. Player experience extraction from gameplay video. *CoRR*, abs/1809.06201, 2018.

[18] Zijin Luo, Matthew Guzdial, and Mark Riedl. Making cnns for video parsing accessible. *CoRR*, abs/1906.11877, 2019.

[19] Henrique Castro Neto and Rita Maria Silva Julia. Ace-rl-checkers: decision-making adaptability through integration of automatic case elicitation, reinforcement learning, and sequential pattern mining. *Knowledge and Information Systems*, 57(3):603–634, Dec 2018.

[20] Kelsey Prena and John Sherry. Parental perspectives on video game genre preferences and motivations of children with down syndrome. *Journal of Enabling Technologies*, 12:00–00, 03 2018.

[21] Sai Rajeswar, Pau Rodriguez, Soumye Singhal, David Vazquez, and Aaron Courville. Multi-label iterated learning for image classification with label ambiguity, 2021.

[22] Mahdyar Ravanbakhsh, Moin Nabi, Enver Sangineto, Lucio Marcenaro, Carlo Regazzoni, and Nicu Sebe. Abnormal event detection in videos using generative adversarial nets, 2017.

[23] Johanna Roettl and Ralf Terlutter. The same video game in 2d, 3d or virtual reality – how does technology impact game evaluation and brand placements? *PLOS ONE*, 13:1–24, 07 2018.

[24] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[25] Lingyun Song, Jun Liu, Buyue Qian, Mingxuan Sun, Kuan Yang, Meng Sun, and Samar Abbas. A deep multi-modal cnn for multi-instance multi-label image classification. *IEEE Transactions on Image Processing*, 27(12):6025–6038, 2018.

[26] Kurt Squire. Video games in education. *International Journal of Intelligent Simulations and Gaming*, 2:49–62, 10 2003.

[27] Adam Summerville, Sam Snodgrass, Matthew Guzdial, Christoffer Holmgård, Amy Hoover, Aaron Isaksen, Andy Nealen, and Julian Togelius. Procedural content generation via machine learning (pcgml). *IEEE Transactions on Games*, PP, 02 2017.

[28] Mingze Xu, Mingfei Gao, Yi-Ting Chen, Larry S. Davis, and David J. Crandall. Temporal recurrent networks for online action detection, 2018.

[29] Manzhu Yu, Myra Bambacus, Guido Cervone, Keith Clarke, Daniel Duffy, Qunying Huang, Jing Li, Wenwen Li, Zhenlong Li, Qian Liu, Bernd Resch, Jingchao Yang, and Chaowei Yang. Spatiotemporal event detection: a review. *International Journal of Digital Earth*, 13(12):1339–1365, 2020.

[30] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[31] Zhi-Hua Zhou, Min-Ling Zhang, Sheng-Jun Huang, and Yu-Feng Li. Multi-instance multi-label learning. *Artificial Intelligence*, 176(1):2291–2320, 2012.