A Sensor Fusion Horse Gait Classification by a Spiking Neural Network on SpiNNaker

Antonio Rios-Navarro(✉), Juan Pedro Dominguez-Morales, Ricardo Tapiador-Morales, Manuel Dominguez-Morales, Angel Jimenez-Fernandez, and Alejandro Linares-Barranco

Robotic and Technology of Computers Lab,
Department of Architecture and Technology of Computers,
University of Seville, Seville, Spain
arios@atc.us.es,
http://www.rtc.us.es

Abstract. The study and monitoring of the behavior of wildlife has always been a subject of great interest. Although many systems can track animal positions using GPS systems, the behavior classification is not a common task. For this work, a multi-sensory wearable device has been designed and implemented to be used in the Doñana National Park in order to control and monitor wild and semi-wild life animals. The data obtained with these sensors is processed using a Spiking Neural Network (SNN), with Address-Event-Representation (AER) coding, and it is classified between some fixed activity behaviors. This works presents the full infrastructure deployed in Doñana to collect the data, the wearable device, the SNN implementation in SpiNNaker and the classification results.

Keywords: SpiNNaker · Spiking neural network · Pattern classification · Horse gait · IMU · Integrate-and-fire · Caffe

1 Introduction

The animal behaviour classification is a challenge for biologists who try to get some patterns from the acts of different animals. There are certain commercial devices that can track animals using GPS and detect their activity level offering some numerical values based on their position. The challenge of the system presented in this paper is to classify the animal behaviour into different patterns, which are previously trained, proposed by biologists. Currently there are some mechanisms like Neural Networks (NNs), SVM (Support Vector Machines), statistical algorithms, among others, that are trained to extract some patterns from large amount of data. These mechanisms are usually implemented on computers but other hardware platforms like [1, 2] have been designed to develop SNNs easily and with better throughput than conventional computers.

The work presented in this paper is part of the Andalusian Excellence project MINERVA, which main aim is to study and classify the wildlife behaviour inside

A.E.P. Villa et al. (Eds.): ICANN 2016, Part I, LNCS 9886, pp. 36–44, 2016.
DOI: 10.1007/978-3-319-44778-0_5

Doñana National Park [3]. To achieve this goal, a hierarchical wireless sensor network capable of gathering and transmitting information from the animals has been deployed and tested inside the park. A low-power collar device attached to the animal, which has an IMU (Inertial Measurement Unit) and a GPS as main sensors, collects the animal information. It sends data throw a ZigBee network of motes to a base station, where this information is classified and uploaded into a cloud server database, using a Wi-Fi link. Biologists can access those reports through a web application, where processed and raw data from animal collars is shown. This paper focuses on the behaviour classification task of horse gaits using SpiNNaker [1], a neuromorphic hardware platform where SNNs can be deployed. The SpiNNaker board is placed in a base station and it receives the collected data from a mini PC (NUC) that interfaces with the ZigBee network using a ZigBee-to-USB bridge board. The raw sensor data is processed by the SNN on SpiNNaker after this information is sent from the collars to the base station and, then, the classification results are sent to the server. This work is focused on horse gaits, but thanks to the configurability of the developed architecture, this classification system can be either extended to other animals or even improved easily by only changing the parameters involved in the training step without the need to catch the animal, which is not possible on the commercial devices that already exist (they only give position and activity level).

The rest of the paper is structured as follows: Sect. 2 describes the hardware platforms involved in this work. Section 3 presents the spiking neural network (SNN) model implemented and its training process. In Sect. 4 the accuracy results of the SNN are shown. Finally, Sect. 5 presents the conclusions.

2 Hardware Platforms

For this work, four different platforms have been designed. They are, from lower to higher operating range: collar, sniffer, base station and central server (see Fig. 1). The collar is the end device, which is attached to the animal. The sniffer device is an easy-to-carry device used by biologists and animal handlers to find a particular animal and obtain its information. Base stations (BS) are placed inside Doñana National Park and they are used as beacons to receive the information transmitted by the collars to classify animal's behavior and send results to a server. Finally, the central server receives the information from BS and stores it into an accessible format through web applications.

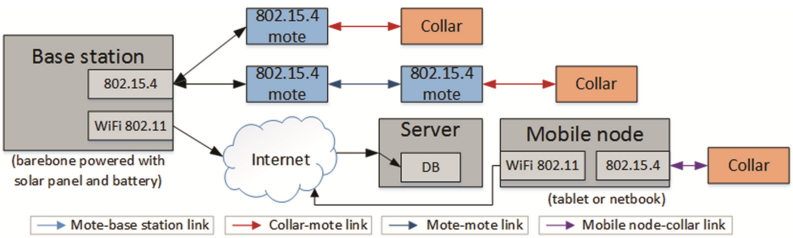

Fig. 1. MINERVA communication topology architecture

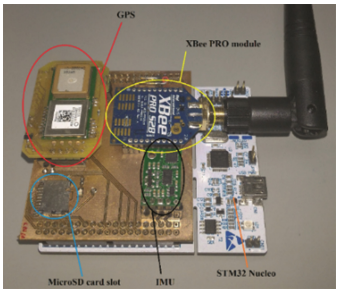

Fig. 2. Collar device prototype

Next, from all these hardware platforms, we will focus on those ones that take part more actively in the SNN classification: collar and BS will be detailed deeply.

2.1 Collar

This device collects information from the environment using multiple sensors. These sensors are a GPS, which gives the position and time; and an inertial movement unit (IMU), which combines 3 different sensors: accelerometer, gyroscope and magnetometer. The IMU has 3 axes for each sensor: (X, Y, Z). These parameters will be taken into account in the SNN for learning and classification steps.

A low power microcontroller is in charge of the measurements. Furthermore, the collar includes a Zigbee module that can transmit data through the network in a wireless way. If the device is out of range from the network, it carries an SD card where the information is always stored; so the animal behavioural information can be accessed later in an offline way, avoiding data loss (see Fig. 2).

2.2 Base Station

The main task of the BS is to receive data packets from collars and to retransmit them to a remote web server via 802.11 Wi-Fi connection. If the collar is out of range, it stores everything in the SD card until it reaches the BS. Moreover, the main impact of these stations is the inclusion of a SNN classifier that allows to know the animal activity. This NN classifier uses the sensors' raw data collected by the collars to obtain the animal behavior, which is later uploaded to the server. Hence, the classification is not done in real time, but in an offline way. This is due to the fact that both the power consumption of the SpiNNaker board and its size are not small enough to be embedded on the collar, which on the other hand leads to implement a simpler firmware on this device.

BS is composed of: the Bridge board, which contains the sensors (i.e. temperature, humidity, luminosity, accelerometer and battery amperimeter) and Zigbee module, which is used to communicate with collars. BS is USB-connected to an Intel NUC [4]. And a battery, a solar panel and a regulator allows the BS to be installed close to wild-animals habitat to charge the battery during daylight (see Fig. 3).

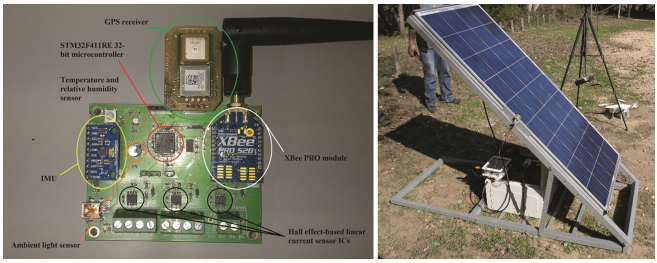

Fig. 3. Bridge board (left) and base station (right)

2.3 Synthetic Spikes Generator (RB-SSG)

A Synthetic Spikes Generator (SSG) will transform digital words into a stream of rate-coded spikes to feed SpiNNaker hardware. This element is necessary for transforming digital sensors information into spikes because the output of these sensors are not spike-coded. There are several ways to implement a SSG as presented in [5]. A SSG should generate a synthetic spikes stream, whose frequency should be proportional to a constant (kSpikesGen) and an input value (x), according to next equation:

$$SSG(x)_{SpikesRate} = k_{SpikesGen} * x \tag{1}$$

The SSG used in this work implements the Reverse-Bitwise (RB) method (details in [6]) for synthetic spikes generation. Figure 4 shows the internal components of the RB-SSG. It uses a continuous digital counter, whose output is reversed bitwise and compared with the input absolute value (*ABS(x)*). If the input absolute value is greater than reversed counter value, a new spike is sent. RB-SSG ensures a homogeneous spikes distribution along time, thanks to reversing bitwise counter output. Since a sensor value can be negative, it is necessary to generate positive and negatives spikes. A demultiplexer is used to select the right output spike port, where selection signal is the input sign (*X(MSB)*). Finally, a clock frequency divider is included to adjust RB-SSG gain. This element will activate a clock enable (*CE*) signal, for dividing the clock frequency, according to a frequency divider signal (*genFD*). So RB-SSG gain ($k_{BWSpikesGen}$) can be calculated as in (2):

$$K_{BWSpikesGen} = \frac{F_{CLK}}{2^{N-1}(genFD + 1)} \tag{2}$$

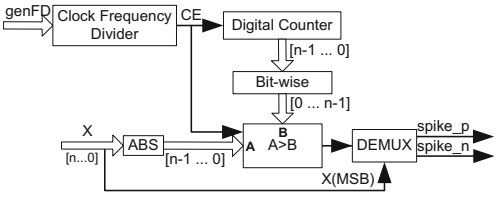

Fig. 4. Reverse Bitwise Synthetic Spikes Generator block diagram.

Where *Fclk* represents system clock frequency, *N* the RB-SSG bits length, and *genFD* clock frequency divider value. These parameters can be modified in order to set up RB-SSG gain according with design requirements.

2.4 Spiking Neural Network Architecture (SpiNNaker)

SpiNNaker is a parallel multi-core computing system designed for modelling very large SNNs in real time. Each SpiNNaker consists of chips (both system architecture and chip are designed by the Advanced Processor Technologies Research Group [7] in Manchester), with eighteen 200 MHz ARM968 cores each. In this work, a SpiNNaker 102 machine, plus PACMAN software was used to implement and test a SNN architecture, presented in the next section. This board has 4 SpiNNaker chips (72 ARM processor cores, where typically are 64 application cores, 4 are monitor cores and there are 4 spare cores) and it requires a 5 V 1A power supply. The control and I/O data is sent through a 100 Mbps Ethernet link. See Fig. 5.

Fig. 5. SpiNNaker 102 machine.

3 Spiking Neural Network Configuration

3.1 Network Architecture

The SpiNNaker platform allows fast and easy SNNs implementation using PyNN [8], which is a Python package for simulator-independent specification of SNN models. PyNN provides a set of different spiking neuron models. However, integrate-and-fire neurons (IF) have been used in this work due to the fact that it is one of the simplest and most widely used models for pattern classification in SNNs. The implemented architecture consists of 3 layers. The input layer receives the stream of spikes (coded in AER [9]) related to the sensor information which was captured with the collar and (converted into aedat files by the SSG) for a specific horse gait [10]. This layer has 9 IF neurons (one per IMU's axis).

Both the hidden and the output layers have the same number of neurons as the desired number of classes to be classified. Three different horse gaits were investigated in this work (motionless, walking and trotting), hence these layers should consist of three IF neurons. Figure 6 shows the SNN architecture implemented on the SpiNNaker board. Two set of connections between consecutive layers of the network are presented: (1) connections between the input and the hidden layers. These connections are trained using

a spike-rate based algorithm which is described in the next section. And (2) connections between the hidden and the output layers. These connections inhibit the unwanted signal from the classification output obtained in the hidden layer.

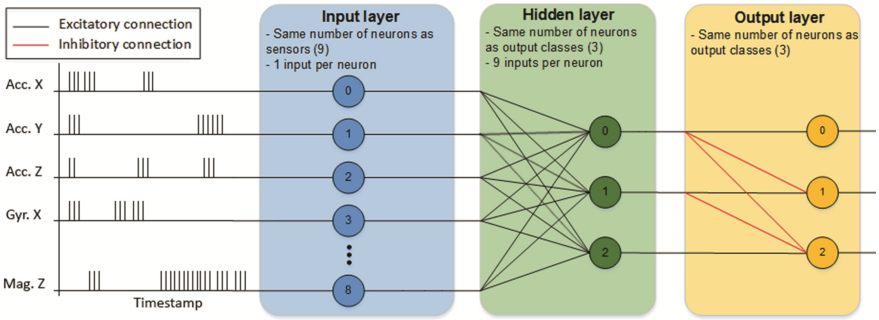

Fig. 6. Spiking neural network architecture using a horse gait aedat file sample as input.

3.2 Training Phase

The weights of the first set of SNN connections are obtained using an offline and supervised spike-rate based training algorithm. These weights are calculated from the normalization of the spike firing activity for each IMU signal using a set of aedat files. These files were generated from a 20-samples frame variance of the raw data obtained in Doñana from the collar while a horse was performing the three different gaits that want to be classified (motionless, walking and trotting). The aim of this training is to obtain the weights of the SNN connections. After that, according to the neuron labels shown on Fig. 6, when the horse is motionless, the only firing neuron in the hidden layer is the number 2; when it is walking, both number 1 and 2 neurons fire; and, in case of trotting, the three of them. Hence, connections between the neurons in the input layer and the neuron 2 of the hidden layer have greater weight values than the connections between the inputs and the number 1, and so on.

The firing rate of a specific degree of freedom ($FRate_{dol[i]}$) is calculated by dividing the number of AER events fired during a certain time period by that time period. Then, the firing rate of a specific sensor (accelerometer, gyroscope or magnetometer) ($FRate_{sensor[i]}$) is the maximum firing rate value of its axes. Finally, $FRate_{dol[i]}$ is normalized using its corresponding $FRate_{sensor[i]}$ value, obtaining (3).

$$FRate_{dol[i]} = \left(\sum AERevents\right)/T_{sample} \tag{3}$$

$$FRate_{sensors[i]} = \max\left(FRate_{dol[x]},\ FRate_{dol[y]},\ FRate_{dol[z]}\right) \tag{4}$$

$$FRate_{dol[i]\_normalized} = FRate_{dol[i]}/FRate_{sensors[i]} \tag{5}$$

As seen in Fig. 7 (left), the highest event rate for each axis is obtained when the horse is trotting (collar in constant motion), hence variance values are greater. However,

setting the weights of the connections with the results obtained from (3) after using these numbers of AER events leads to a firing output in the hidden layer that is the opposite of what was expected. To solve this problem, the normalized firing rate values are inverted (see Fig. 7 right) and used as weights for the connections between input and hidden layers.

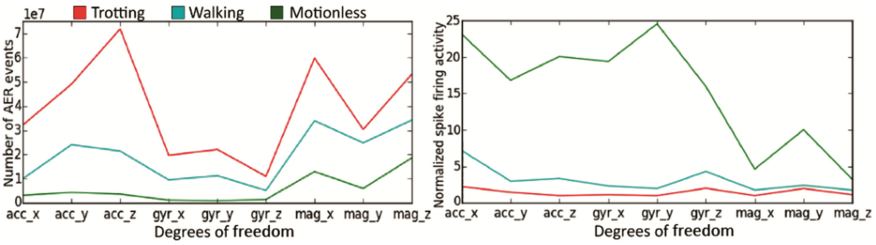

Fig. 7. AER events fired (left) and normalized activity (right) for each IMU degree of freedom

The connections between hidden and output layers are configured such as the first neuron of the hidden layer inhibits the last two neurons of the output layer. However, the second neuron of the hidden layer only inhibits the last neuron of the output layer. With this configuration, when the horse is performing one of the gaits that want to be classified, only its corresponding output neuron fires.

4 Results

The information required for training and testing the SNN has been collected with the collar placed on a horse. This specie was chosen due to the weight of the collar prototype. For the testing scenario, the collar collects information continuously from on-board sensors and send it to a computer application. The information is stored in different files depending on the behaviour.

The information is obtained by a biologist managing the animal while a user captures each behaviour in different files: 6000 samples for each behaviour have been collected. Next, the dataset is pre-processed by calculating the variance using 20-sample windows. Finally, 300 samples per behaviour are used for training (200) and testing (100) the SNN after converting them to spikes. Figure 8 shows the accuracy results of this test.

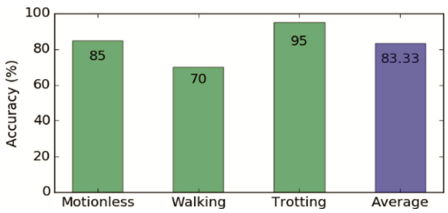

Fig. 8. Accuracy results for the SNN SpiNNaker implementation

The network proposed in this paper has been compared with a modification of the LeNet5 [11] ConvNet, which has been trained and tested using Caffe [12] (Convolutional Architecture for Fast Feature Embedding). This network is well known for its high accuracy results for image recognition, so data samples have been converted to frames. The network is fed with these frames, using the same train and test ratio of the whole dataset. Each frame has been composed as Fig. 9-left shows.

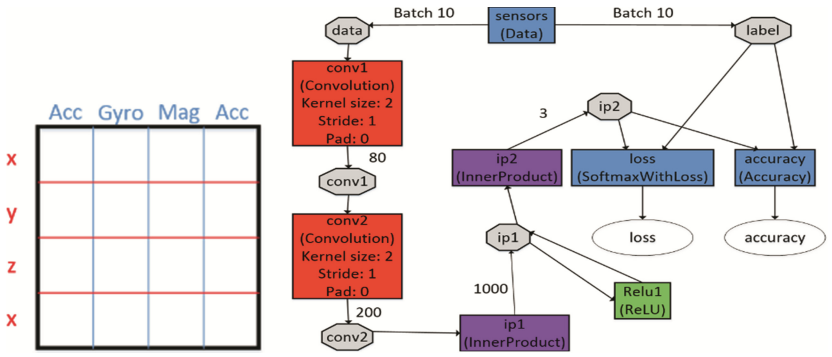

Fig. 9. (left) Frame structure using sensor variance. Column: IMU sensor, Row: coordinate. (right) Network topology from Caffe

Figure 9-right shows the modified LeNet5 network where pooling operations have been removed in both convolutional stages because input frames have low pixel resolution. After training and testing this network, an 81.2 % average accuracy value is obtained.

5 Conclusions

In this manuscript, a novel SNN for horse gait classification was implemented and tested using SpiNNaker. For that purpose, a collar with a low-power microcontroller and a 9-axis IMU has been developed along with a desktop application for collecting the data. This data was obtained in Doñana National Park from different horses in three different seasons of the year. After transforming the collected sensor information into spike-streams using the RB-SSG, the SNN was trained. And then, several classification tests were performed on the SpiNNaker board, obtaining an 83.33 % accuracy on average. These tests were also performed using a modified LeNet ConvNet implemented in Caffe to compare the results, obtaining an 81.2 % accuracy. The SpiNNaker board has allowed modeling, implementing and testing a SNN for this purpose in an easy and fast way, proving its versatility and efficiency when deploying SNNs in hardware platforms.

Acknowledgement. The authors would like to thank the Advanced Processor Technologies (APT) Research Group of the University of Manchester for instructing us on the SpiNNaker platform on the 5th SpiNNaker Workshop. This work is supported by the Spanish government

grant BIOSENSE (TEC2012-37868-C04-02) and by the excellence project from Andalusian Council MINERVA (P12-TIC-1300), with support from the European Regional Development Fund.

References

1. Painkras, E., et al.: SpiNNaker: A 1-W 18-core system-on-chip for massively-parallel neural network simulation. IEEE J. Solid-State Circ. **48**(8), 1943–1953 (2013)
2. Merolla, P.A., et al.: A million spiking-neuron integrated circuit with a scalable communication network and interface. Science **345**(6197), 668–673 (2014)
3. Doñana National Park: http://whc.unesco.org/en/list/685
4. Intel NUC. http://www.intel.com/content/www/us/en/nuc/nuc-kit-d54250wykh.html
5. Gomez-Rodriguez, F., Paz, R., Miro, L., Linares-Barranco, A., Jimenez, G., Civit, A.: Two hardware implementations of the exhaustive synthetic AER generation method. In: Cabestany, J., Prieto, A.G., Sandoval, F. (eds.) IWANN 2005. LNCS, vol. 3512, pp. 534–540. Springer, Heidelberg (2005)
6. Paz, R., et al.: Synthetic retina for AER systems development. In: AICCSA 2009, pp. 907–912 (2009)
7. A.P. Techonologies Research Group: http://apt.cs.manchester.ac.uk/projects/SpiNNaker
8. Davison, A.P.: PyNN: a common interface for neuronal network simulators. Front. Neuroinform. **2**, 11 (2008)
9. Sivilotti, M.: Wiring Considerations in analog VLSI Systems with Application to Field-Programmable Networks. California Institute of Technology, Pasadena CA (1991)
10. Harris, S.E.: Horse Gaits. Balance and Movement (1993)
11. LeCun, Y., et al.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2323 (1998)
12. Jia, Y., et al.: Caffe: Convolutional Architecture for Fast Feature Embedding. In: Proceedings of the ACM International Conference on Multimedia, pp. 675–678 (2014)