**Automatization of livestock health checks with the use of algorithms**

| Julian Andres Romero  Universidad Eafit  Colombia  jaromeroh@eafit.edu.co | Juan Avendaño  Universidad Eafit  Colombia  jeavendanc@eafit.edu.co | Simón Marín Universidad Eafit Colombia smaring1@eafit.edu.co | Mauricio Toro  Universidad Eafit  Colombia  mtorobe@eafit.edu.co |
| --- | --- | --- | --- |

# **ABSTRACT**

Animals are an important part of life from antiquity, they provide transportation, draught power, fuel, clothing as well as proteins. That’s why it’s considered that their health is very important in order to have sustainable livestock production. Evidently, these problems can be solved by the inclusion of new technology to livestock farming. New technology like machine learning that can help in the recognition of sick animals, with the purpose of treating them as fast as possible, all helped by compression algorithms and deep learning

# **1. INTRODUCTION**

Nowadays, technology has taken power over the world. We have seen how even the most traditional jobs such as farming and animal breeding have radically changed, with workers improving every day to be much more productive and make their job easier. Following this idea, algorithms and machine learning are a great inclusion to this industry, because using them we can analyze big amounts of data and know the current state of the cattle, all thanks to compression algorithms and image recognition made through machine learning.

# **1.1. Problem**

Livestock farming is fundamental for the diet of today’s society, but the constantly increasing population is demanding more livestock each day, and that is why it is fundamental to make the process as effective and productive as possible. Diseases and injuries have always been one of the biggest problems for breeders, as an unhealthy animal is basically an economic loss, and with society demanding better living conditions for animals, it is now mandatory for breeders to look after the health of their animals. With the automatization of health checks for animals, meat costs could decrease, as the losses for sick animals would be reduced, and the life quality of animals would improve because they could be treated on time.

**1.3 Article structure**

In what follows, in Section 2, we present related work to the problem. Later, in Section 3, we present the data sets and methods used in this research. In Section 4, we present the algorithm design. After, in Section 5, we present the results. Finally, in Section 6, we discuss the results and we propose some future work directions.

**2. RELATED WORK**

In the ensuing paragraphs, we exhibit four related works related to the problem that we treat in this article, showing a variety of solutions to this type of problem and in the context of PLF.

**2.1. Sensors, big data and machine learning in modern animal farming.**

To improve feeding efficiency and animal health in livestock farming In order to solve complex problems such as the nutrient excretion into the environment, the identification of functional limiting factors, optimal nutrient composition, among others, the authors use mechanistic models, which needs large volumes of diverse datasets with local weather data, air quality data, and visual data. This is where the sensors make a big role, as sensors are a device which measures chemical, biological and physical data. The use of big data offers the possibility to store a vast amount of data on a remote server. Advanced AI and ML algorithms can make use of this extensive data to analyze, predict and notify farmers in case there is something abnormal.

**2.2 Using Edge Analytics to Improve Data Collection in Precision Dairy Farming**

The use of Wireless Sensor Networks (WSN) in precision farming improves agricultural productivity and sustainability, as WSN facilitates the collection of farm data, which is used for better monitoring and understanding of the farm processes. However, only a few WSN systems have been put into practice due to the lack of infrastructure in a typically remote farm environment. For this reason, this paper presents a WSN prototype system for data collection in a dairy farm with no transmission to the cloud storage in a timely manner. To achieve that, it is necessary to optimize memory usage for sensor devices, so data compression is key in this matter. The analyzed compression techniques were Data Transform Coding (DTC), Data Source Coding (DSC) and Compression Sensing (CS) for use in dense sensor networks, and Lightweight Temporal Compression (LTC) and Two-Modal Transmission (TMT) for sparse sensor networks

**2.3 Internet of Things in arable farming: Implementation, applications, challenges and potential**

## IoT has recently been gaining momentum in the farming industry as it can fulfill the urgent necessity for interoperability across brands, scalability and traceability. Multiple applications can be derived from the implementation of IoT in arable farming. These applications can always be conceptualized into three IoT layers: Application layer, Network layer and device layer. One of the challenges when implementing IoT in arable farming is data heterogeneity, as The diverse data sources and sensor manufacturers imply the use of different unit systems, data structures and nomenclatures in different data formats, which result in reduced syntactical and semantic interoperability among IoT environments.

**2.4 The use of infrared images to detect ticks in cattle and proposal of an algorithm for quantifying the infestation**

The article presents a study on the use of low resolution infrared images to detect ticks in cattle. Ticks cause significant economic losses both as a blood sucking parasite and as a vector of pathogens and toxins. In order to maximize the effectiveness of control methods, the infestation must be quantified, and one of the technologies that has the potential to replace the traditional manual method is the infrared thermography (IRT). For this study, One hundred animals of the Brangus breed (heifers and young bulls) were used in the image capture, with approximately half being naturally infested and half artificially infested with *Rhipicephalus* (*Boophilus*) *microplus* ticks. The body parts selected for imaging were the neck and hind end below the tail, with a frame of 20 × 20 cm delimiting the region of interest. The study ended up with the conlusion that infrared thermographic images have limited potential to be used in the detection of ticks in cattle, enabling just a rough estimate for the degree of infestation, which can be obtained either visually or automatically using the algorithms.

## **3. MATERIALS AND METHODS**

In this section, we explain how the data was collected and processed and, after, different image-compression algorithm alternatives to improve animal-health classification.

## **3.1 Data Collection and Processing**

We collected data from Google Images and Bing Images divided into two groups: healthy cattle and sick cattle. For healthy cattle, the search string was “cow”. For sick cattle, the search string was “cow + sick”.

In the next step, both groups of images were transformed into grayscale using Python OpenCV and they were transformed into Comma Separated Values (CSV) files. It was found out that the datasets were balanced.

The dataset was divided into 70% for training and 30% for testing. Datasets are available at [https://github.com/mauriciotoro/ST0245-Eafit/tree/master/proyecto/dataset](https://github.com/mauriciotoro/ST0245-Eafit/tree/master/proyecto/datasets)s .

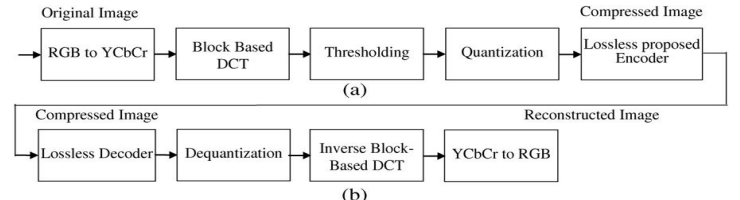
Finally, using the training data set, we trained a convolutional neural network for binary image-classification using Google Teachable Machine available at <https://teachablemachine.withgoogle.com/train/image>.

## **3.2 Lossy Image-compression alternatives**

## In what follows, we present different algorithms used to compress images.

**3.2.1 JPEG COMPRESSION**

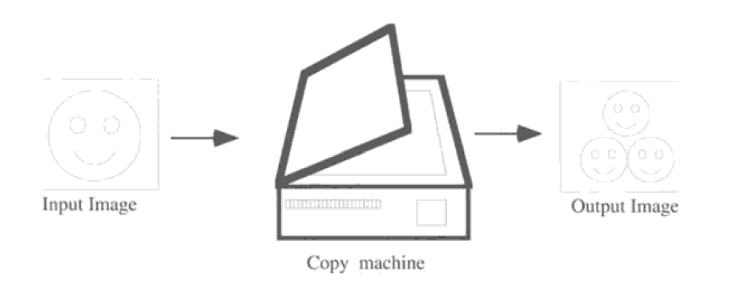
JPEG compression works using the discrete cosine transform, and it reduces the image size, losing the least important information. That’s why it is considered lossy-compression, because the original and compressed images aren’t completely the same.



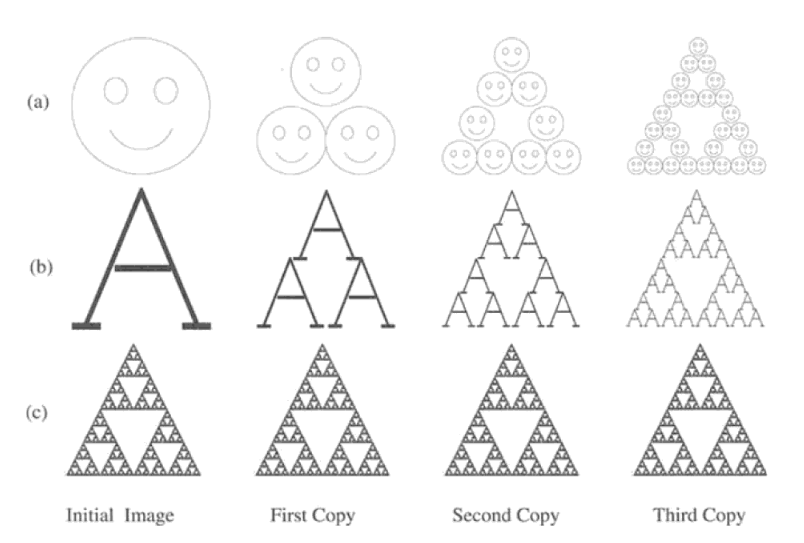
When we talk about a good implementation o f this algorithm, we can say that it has a complexity of *O(n)* where *n* is the number of pixels.

**3.2.2 Fractal image compression**

Fractal image compression consists of reducing an image to be copied and reproducing it three times, as shown in the image below.



After this, the output image is given as input and the process is repeated. As a result, a pattern is formed as seen in the next image.

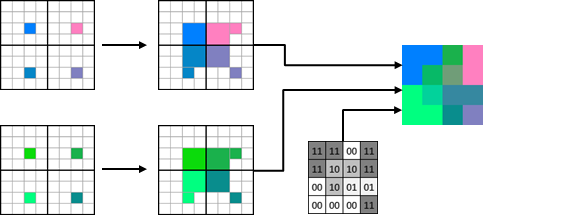


This method is often used for textures and natural images, because parts of the image resemble other parts of the same image.

The complexity of processing an image of size *n* by *n* is O(*n*³)

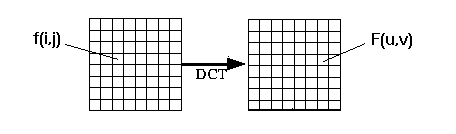
**3.2.3 S3 Texture Compression**

This algorithm splits the image into 4x4 blocks to perform compression in each block separately. Then the 16 color values of each block are approximate to only four and the with two, as shown below.



**3.2.4 The discrete cosine transform (DCT)**

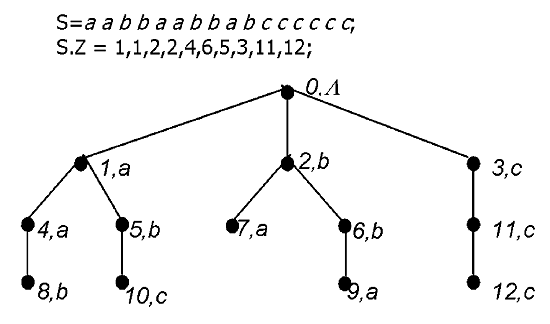
The discrete cosine transform (DCT) helps separate the image into parts (or spectral sub-bands) of differingimportance (with respect to the image's visual quality). It transforms a signal or image from the spatial domain to the frequency domain



## **3.3 Lossless Image-compression alternatives**

## In the following section, we present different algorithms used to compress images in the lossless way possible.

**3.3.1 LZW (Lempel-Ziv-Welch)**

This algorithm, simple to implement and with very high throughput in hardware implementation, works reading a sequence of symbols, grouping the symbols into strings, and converting the strings into codes, as code takes up less space than strings, generating compression. To generate this code, a dictionary keeps a correspondence between the longest encountered words and a list of code values, replacing the words with their corresponding codes. The complexity of this algorithm is O(n), as each byte is only read once and the complexity of the operation for each character is constant

**3.3.2 Huffman Coding**

Huffman Coding is an algorithm that first creates a tree using the frequencies of the character and then generates code for each character. Decoding is done the same way, using the same tree. Taking a String as an example, to compress that string using Huffman Coding it is needed to follow the next steps:

1. Calculate the frequency of each character in the string.

2 Sort the characters in increasing order of the frequency. These are stored in a priority queue Q

3.Make each unique character as a leaf node.

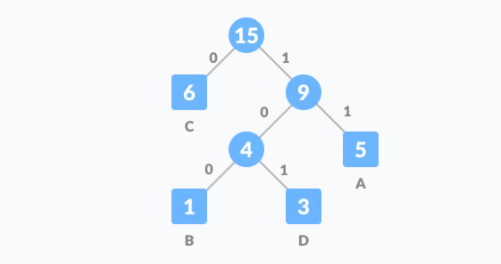
4.Create an empty node z. Assign the minimum frequency to the left child of z and assign the second minimum frequency to the right child of z. Set the value of the z as the sum of the above two minimum frequencies.

5. Remove these two minimum frequencies from Q and add the sum into the list of frequencies (\* denote the internal nodes in the figure above).

6. Insert node z into the tree.

7. Repeat steps 3 to 5 for all the characters.

8. For each non-leaf node, assign 0 to the left edge and 1 to the right edge.

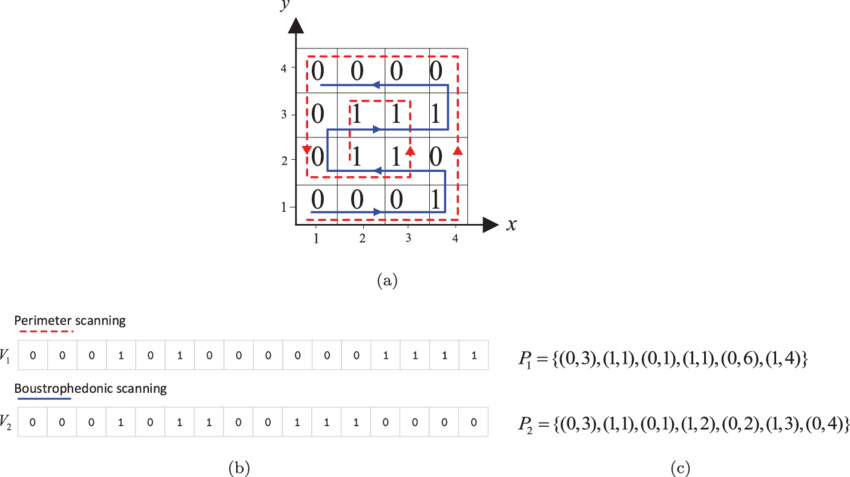


**3.3.3 RLE compression algorithm**

Run-length encoding is a simple way of compressing information, running sequences with the same value occurring many consecutive times, storing only a single value and its count. For example, in a screen containing black text on a white background, there will be many long runs of white pixels in the black space and many short runs of black pixels within the text. A run-length encoding data compression algorithm applied to the following hypothetical scan line

XXXXXXXXXXXXXXZXXXXXXXZZXXXXXXXZXXX

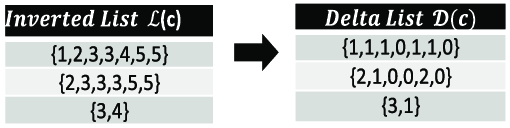
Would render it as 14X1Z6X2Z7X1Z3X, a sequence of twenty X’s, one Z, six X’s, two Z’s, seven X’s, one Z and three X’s



The time complexity in this algorithm is O(n), where n is the length of the input.

**3.3.4 Delta encoding**

The Delta encoding algorithm has as objective to get only the bytes that have been modified since the last version of the file, reducing considerably the size of the file. Once the file is compressed, it can be obtained from the original file by getting the reference version of the file and the file generated by the algorithm.



# **REFERENCES**

1. Neethirajan, S. The role of sensors, big data and machine learning in modern animal farming
2. Bhargava, K. Using Edge Analytics to Improve Data Collection in Precision Dairy Farming
3. Barbedo, J.G.A The use of infrared images to detect ticks in cattle and proposal of an algorithm for quantifying the infestation
4. Henriksen, V.A Internet of Things in arable farming: Implementation, applications, challenges and potential

5. Izhak Baharav, Ben Bielefeld, Roger D. Boss. Fractal image compression, Theory and application. Springer-Verlag, New York, 1995, 340.

6. Dietmar Saupe. Breaking the Time Complexity of Fractal Image Compression. Universitat Freiburg.

7.. A.M.Raid, W.M.Khedr, M. A. El-dosuky, Wesam Ahmed. Jpeg Image Compression Using Discrete Cosine Transform - A Survey. Mansoura University, Faculty of Computer Science and Information System