Final Proyect
Supervised Learnig solution

Joseph Jesus Aguilar Rodriguez, *1809003*, Gabriela Elizabeth Avila Chan, *2009003*,
Leonardo Cañamar Amaya, *2009018*, Marco Alejandro González Barbudo, *1909073*,
Diego Armando May Pech, *2009092*

***Abstract*—This document presents a comprehensive code solution aimed at ensuring the safety of Gupy, the beloved mascot of Universidad Politécnica de Yucatán. By employing supervised learning techniques, the project focuses on Gupy detection from images, utilizing datasets of cats to precisely identify Gupy. The code implements machine learning to develop an algorithm capable of safeguarding Gupy, promoting his well-being, and addressing the community's concerns.**

***Index Terms*—Artificial Intelligence, Machine Learning, Learning algorithms, Supervised learning, Image recognition, Code implementation.**

I. Introduction

IN light of recent events surrounding the unfortunate incident in which the beloved Yucatan Polytechnic University mascot, Gupy, was injured, there is an urgent need to implement enhanced safety measures to ensure the protection and well-being of Gupy. This paper proposes a comprehensive code solution that leverages supervised learning techniques for a future application in robotics to mitigate potential risks and ensure the continued security of Gupy. The aim is to implement a Perceptron algortihm to train it with the objective of detecting and distinguishing Gupy from other cats.

The project first part focuses on Gupy detection from images, utilizing datasets of cats to precisely identify him within the university environment. By implementing a machine learning algorithm, this initiative seeks to address the concerns of the campus community and provide a proactive approach to Gupy's safety.

II. Development

In order to achieve the goal, a machine learning algorithm was designed and trained to predict if the output picture was Gupy or not. Developing a Jupiter Notebook that contains a code it is enhance the training of the algorithm to perform this labor.

A. Dataset

This Perceptron[1] was fed a carefully curated self-made dataset, encompassing a diverse range of photos featuring Gupy alongside images of other traditional cats. I This deliberate inclusion of various cat breeds aimed to provide the model with a rich and varied dataset, facilitating the learning process for the algorithm to accurately identify the distinct features of Gupy. [1]

The dataset compilation involved a comprehensive selection of images capturing Gupy in different poses, lighting conditions, and backgrounds, ensuring that the model could generalize well across various scenarios. Additionally, the inclusion of images of other cats served to challenge the model and enhance its ability to discriminate between Gupy and other feline counterparts. In the following sections, a detailed examination of the algorithm and its outcomes will be presented.

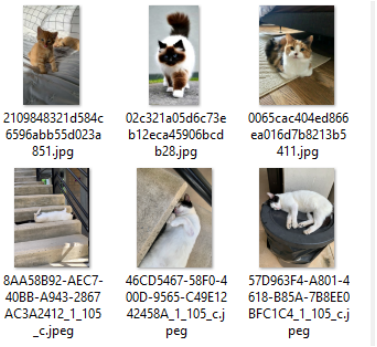

Fig. 1. Preview of the dataset made for the model.

B. Model Pre-Preparation

This section aims to provide an overview of the design and training process of the Perceptron for the accurate Gupy detection.

To begin with, `zipfile` module was used to extract the contents of a ZIP file. Then, the path to the ZIP file and the directory were specified. The ZIP file is opened and the all the elements are extracted archive to the specified destination directory using `zip_ref.extractall(directorio_destino)`. Finally, a message indicating the successful extraction of files to the specified destination directory is printed.

```
import zipfile

archivo_zip = "/content/gupy_dataset-
20231104T191923Z-001 (1).zip"
directorio_destino = "/content"

with zipfile.ZipFile(archivo_zip, 'r')
as zip_ref:
```

```
    zip_ref.extractall(directorio_destino)

print(f"Archivos descomprimidos
en {directorio_destino}")
```

Then, all the necessary libraries to process, train, test the model and plot the prediction results.

```
import pandas as pd
import numpy as np
from sklearn.model_selection
import train_test_split
from sklearn.preprocessing
import StandardScaler
from sklearn.linear_model import Perceptron
from sklearn.metrics import accuracy_score
import cv2
import random
import os
from PIL import Image
import matplotlib.pyplot as plt
```

The path for the folder containing images is set by defining the variable `carpeta_imagenes` and assigning it in the path '/content/gupy_dataset-20231104T191923Z-001/gupy_dataset', indicating the folder where the images are located. After that, the variable `csv_path` is defined and assigned in the path '/content/labels_my-project-name_2023-11-04-03-14-11.csv', indicating the path to the CSV file containing the dataset. Finally it uses `pd.read_csv()` from the `pandas` library to read the CSV file specified by `csv_path`. The resulting dataset is stored in the variable `data`. The argument `encoding='latin-1'` is used to handle any special characters in the CSV file.

```
carpeta_imagenes = '/content/gupy_dataset
-20231104T191923Z-001/gupy_dataset'

csv_path =
'/content/labels_my-project-name.csv'
data = pd.read_csv(csv_path,
encoding='latin-1')
```

A process to perform the label replacement and feature extraction from a dataset is done. The `map()` method is used on the 'label_name' column of the `data` DataFrame. This operation replaces the labels 'GUPY' with 1 and 'NONGUPY' with 0, effectively converting categorical labels into numerical representations.

A new variable `X` is created by extracting specific features from the DataFrame. The features selected are the bounding box coordinates: 'bbox_x', 'bbox_y', 'bbox_width', and 'bbox_height'. These features are organized into a NumPy array using `.values` for further us in the model.

```
data['label_name']=data['label_name'].map
({'GUPY': 1, 'NONGUPY': 0})

X = data[['bbox_x', 'bbox_y',
'bbox_width', 'bbox_height']].values
```

The label ('label_name') is extracted from the dataset and assigns them to the variable `y`.Then, the dataset is split into training and testing sets using `train_test_split`. The parameter `test_size=0.2` specifies that 20% of the data will be used for testing, and `random_state=42` ensures reproducibility.

To finish with the preparation of the data to be used by the model. The features are normalized by using `StandardScaler`. The training set (`X_train`) is fitted and transformed, and the testing set (`X_test`) is transformed using the same scaler.

C. Model Design

Once the dataset is ready to be used, the Perceptron model is created with a maximum of 1000 iterations and a random seed for reproducibility. The model is trained using the normalized training data (`X_train` and `y_train`). With the trained model, it makes predictions on the testing set (`X_test`) using the trained Perceptron model.

Finally, the accuracy of the model is computed by comparing the predicted labels (`y_pred`) with the actual labels in the testing set (`y_test`). The accuracy of the Perceptron model as a percentage is also printed.

```
y = data['label_name'].values

X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

perceptron = Perceptron(max_iter=1000,
random_state=42)

# Train the Perceptron
perceptron.fit(X_train, y_train)

# Make prediction
y_pred = perceptron.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Precisión del Perceptrón:
{:.2f}%".format(accuracy * 100))
```

D. Model Evaluation

To evaluate the model, a random image from the dataset will be selected from a row index from the dataset (`data`) and retrieves the corresponding image name from the 'image_name' column. With the selected image, a path of the selected image by joining the image folder path (`carpeta_imagenes`) with the image name (`imagen_nombre`).

Then, the bounding box features is extracted of the randomly selected image from the dataset. The features include 'bbox_x', 'bbox_y', 'bbox_width', and 'bbox_height'. Also, the bounding box features are normalized of the randomly

selected image using the previously defined scaler (`scaler`). Finally, the trained Perceptron model (`perceptron`) is used to make a prediction based on the normalized bounding box features (`bounding_box_normalizada`).

```
fila_aleatoria = random.choice(data.index)
imagen_nombre = data.at[fila_aleatoria,
'image_name']

ruta_imagen = os.path.join
(carpeta_imagenes, imagen_nombre)
bounding_box = data.iloc
[fila_aleatoria, 1:5].values
bounding_box_normalizada =
scaler.transform([bounding_box])

prediccion = perceptron.predict
(bounding_box_normalizada)
```

The binary prediction is mapped to categorical labels to display the information about the prediction of the image. For that, the first thing to do is to check if the predicted value (`prediccion`) is equal to 1. If true, assigns the label "GUPY" to the variable `resultado`; otherwise, assigns "NONGUPY". This step is a mapping of the numerical prediction to human-readable categories.

Then, a message indicating the image name (`imagen_nombre`) and the predicted label (`resultado`) is printed. This provides information about the prediction for the displayed image.

To do so, it uses the `Image.open()` function from the `PIL` library to open the image located at `ruta_imagen`. The image is then displayed using `plt.imshow()` from `matplotlib`. The `plt.axis('off')` command is used to hide the axes, providing a cleaner image display.

```
if prediccion == 1:
    resultado = "Gupy"
else:
    resultado = "NONGUPY"

print("La imagen", imagen_nombre,
"es:", resultado)

imagen = Image.open(ruta_imagen)
plt.imshow(imagen)
plt.axis('off')
plt.show()
```

III. Results

The Perceptron model trained for Gupy detection has yielded promising results, demonstrating an accuracy of 97.80%. This outstanding performance underlines the effectiveness of the model in the specific task of identifying the presence of Gupy in images.

A. Performance Metrics

Accuracy: 97.80

B. Examples of Predictions

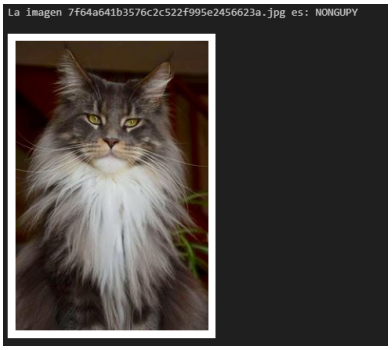

Fig. 2. NONGUPY was not predicted.

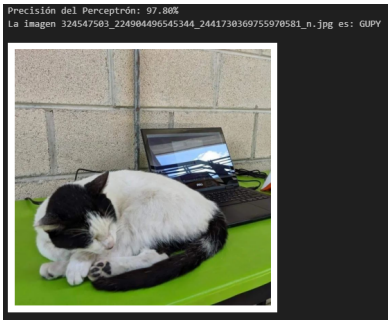

Fig. 3. Gupy was predicted.

These examples illustrate the model's ability to distinguish between images that contain Gupy and those that do not, supporting the high accuracy reported.

The performance of the model is not limited to just the overall accuracy metric, as it has been extensively evaluated on various images, thus ensuring its robustness and ability to generalize to new data.

IV. Conclusion

In conclusion, this document effectively demonstrates a robust and innovative approach to enhance the safety of Gupy, the mascot of Universidad Politécnica de Yucatán, through a machine learning solution. Employing a Perceptron Algorithm, the project successfully developed an algorithm for Gupy detection from images, distinguishing him from other cats with considerable accuracy.

The process involved creating a diverse dataset featuring Gupy and various other cats, ensuring comprehensive learning and generalization capabilities of the model across different scenarios. The implemented Perceptron model, after being trained on this dataset, showcased promising results, reflected in its high accuracy in identifying Gupy.

Moreover, the model was not only capable of accurately identifying Gupy but also provided insights into its performance through practical evaluation using random images from the dataset. This evaluation demonstrated the model's ability to effectively apply its learned features in real-world conditions.

Overall, this project not only addressed the immediate need for Gupy's safety but also set a precedent for using machine learning in mascot security and similar applications. The successful implementation and positive outcomes of this project underscore the potential of machine learning in enhancing safety and security measures in various domains.

APPENDIX

A. GitHub Repository: Labels and Jupyter Notebook

In the following GitHub repository you can find the .csv that contains the labels corresponding to the images of the dataset as well as the Jupyter Notebook in which the perceptron model was developed.

https://github.com/Maages09/SupervisedLearning.git

B. Dataset

In the following link you can find the dataset corresponding to the images.

https://upy-my.sharepoint.com/:u:
/g/personal/1909073_upy_edu_mx/
EWrk09gIIYVKu0yPHPmOI20Bg7fdfhYwZfVeQq_
Q3a7YuA?e=Z8QdYd

REFERENCES

[1] Alpaydin, E. (2014). The Perceptron. In Introduction to Machine Learning (3rd ed., pp. 267-316). MIT Press