Real Time visual tracking with RRN

Final Project

CS577 - Deep Learning

Guillermo Lopez-Areal (#A20520157)
Eneko Gonzalez (#A20520157)

Abstract

One of the most fundamental and important tasks in computer vision is object detection. Its job is to identify all the relevant items in the image, as well as their category and position. Applications include face detection, pedestrian detection, and vehicle detection. Real-time tracking rather than object recognition and tracking in static images, is preferred in some applications, such as robots and surveillance. Nevertheless, a first step in this field has been to track and recognize objects in static images, and then it has been escalated to video object tracking. Deep learning techniques have recently been demonstrated to perform better than prior state-of-the-art machine learning techniques in several domains, with computer vision being one of the most notable examples.

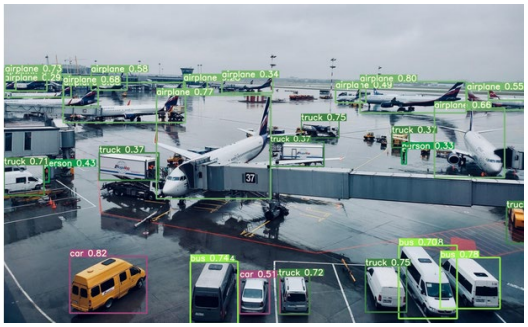
1. Introduction & Problem Statement

Being one of the most demanding and competitive fields in nowadays society, artificial intelligence is a field that is unquestionably on the rise, and a field which within not too many years, will become a part of everybody's life, specially, the field of computer vision which is one of the most promising.

The study of computational techniques to assist computers in comprehending and interpreting the content of digital

photos and videos is known as "computer vision".

The goal of computer vision is to enable computers to perceive and comprehend visual data coming from cameras or sensors.



Computer vision applications are used in a wide range of industries, ranging from security and medical imaging to manufacturing, automotive, agriculture, construction, smart city, transportation, and many more.

The project consists of the development of a deep learning model that has a real application on life circumstances. This project will deal with what is one of the most important fields in robotics and computer vision, the tracking and recognition of objects in images.

Existing algorithms can track objects well in controlled environments. However, they usually fail in the presence of significant variation of the object's appearance or surrounding illumination. Additionally, those able to handle this situation are usually very heavy algorithms with low levels of efficiency. The handling of non-stationary data, not only the target object, but also, the background, is what the dataset environment for object recognition will be.

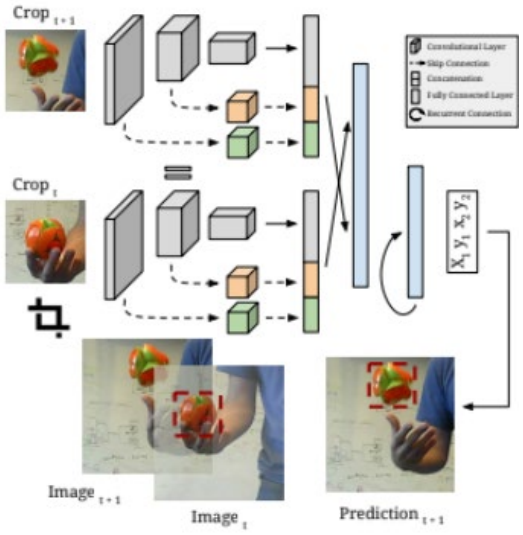
The main goal of our project is to estimate the future position and recognition of a visual target. Always, trying to implement the most modern and advanced techniques.

2. Fundamentals of the project approach

To face the complexity of the problem, clever solutions and techniques must be used. For this purpose, the well proposed solution from Daniel Gordon, Ali Farhadi and Dieter Fox will be implemented [2].

On their paper, they describe properly the real scenario and situation of the object tracking on the moment. Based on thar scenario they proposed an approach to the problem based on Recurrent Neural Networks.

More specifically, they present a new type of architecture to face the object tracking in Deep Learning techniques. This architecture is described as a combination of convolutional networks, RNN networks and fully connected layers.



The main idea is to use the convolutional layers for the object appearance and location, the recurrent neural networks for being able to remember the object form and the movements and finally a

fully connected layer to work on that information and being able to present a location of the object.

The network is of regression type as it will estimate the top left and top right position coordinates. This will allow to estimate the size of the rectangle that will determine the position of the object in the image.

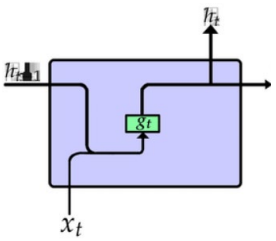
2.1. Theoretical base

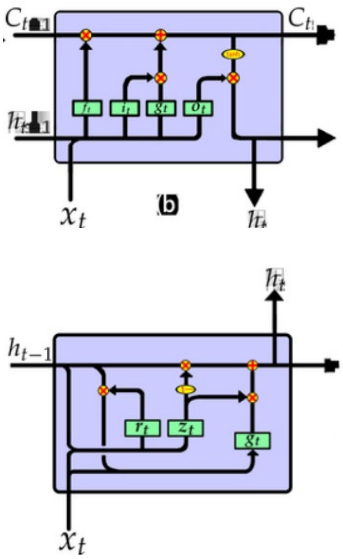
As mentioned before, the architecture is a new approach for this type of problems and the idea is based on the application of regression recurrent networks.

Why the use of Recurrent Neural Networks, in a matter of object tracking? As the only neural network with an internal memory, RNNs are a strong and reliable type of neural network and one of the most promising ones currently being used.

RNNs can accurately forecast what will happen next because of their internal memory, which helps them to retain key details about the input they received.
Because they can keep the feature information of past data in the internal state, RNNs are mostly employed to process sequence data (i.e., memory).

As we can see in the figure below, there are multiple types of RNN's, the first picture represents a conventional vanilla RNN, the second one represents a LSTM (Long short-term memory network) and the third one represents a Gated recurrent unit (GRU)





RNNs, such as LSTM, are used for cases where the data includes temporal properties, e.g., time series, and where the data is context sensitive.

3. Proposed solution

The matter at hand is considerably complicated, therefore, the methodology chosen to address the problem has been to divide and conquer. Based on this idea the solution has been proposed in three parts and models.

3.1. Object Detection system

First idea in the development is the developed what is the most basic part in complex object recognition, the detection of the object.

This model consists on identifying the position of a single object on an image and being able to determine its size and location. To do so, the project loads a set of images. Each of them having a single object to identify and a set of coordinates indicating the rectangle that identifies the object.

In this way, and with a network based simply on fully connected layers, the

model can estimate the actual position of an object in the image.

In this model, each image and data are treated independently, so no relation between the difference instances is created. Not at least on a sequential way.

3.2. Object tracking with RNN

Second developed model goes one step forward. In this case, the object is not only detected, but it is tracking during a set of images that represent a movement.

This Object tracking approach introduces several differences on how the model is built and how the data is treated. For this case, the data will be introduced in a sequential way and will be treated this way too.

To do so, the network introduces Convolution and RNN layers to the previous fully connected ones. In this way, the image can be better processed, and relations can be established from previous outputs and the new given inputs. Concretely, ConvLSTM layers have been deployed which can predict the future placement of an object based on earlier data.

This approach is the one that is the closest to the original approach proposed by the paper the project relies on.

3.3. Third party libraries (OpenCV)

For the third step, one step forward has been done. This time the full potential of the technology has been tested.

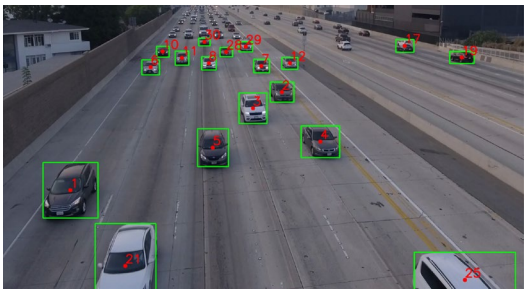
This kind of model are of very high level of complexity. This is way developing a fully functional system is a hard task and requires more data and computational power than we have access to.

But there exit lots of already trained models that allow normal users to have access to this technology.

To prove it we have implemented one of the multiple existing libraries: OpenCV. It is one of the most known, in the field of computer vision. Additionally, it also has functionalities for running deep learning inference as well.

One of the libraries we have tried is OpenCV DNN module, and we have checked with a video how well the object recognition went.

The video consists of multiple cars in the highway of Los Angeles, and the software predicts frame by frame which cars are detected with a certain precision.



4. Description of the data.

Data is a key element in every Artificial Intelligence field development. For this reason, the project described in this document required concrete data.

As mentioned before, the complexity of the topic unable the fully functional development and part of it is the complexity of the data.

So, the made approach to simplify the concepts has suppose the requirement of personalized data. This data has been created to being able to achieve to goals of the project correctly.

Concretely, a visual tracking dataset generator has been created.

4.1. Data set generator

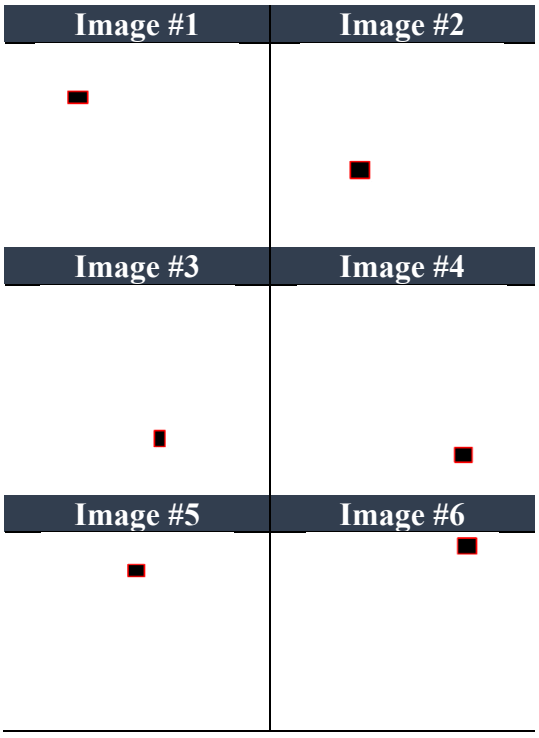
The dataset generator is a code that generates a random or sequential set of images.

The images can be of any given size and can introduce one or more objects of a defined size.

Additionally, the number and location of the images can be defined.

To complete the functionalities of the dataset generator it also creates a .csv file that includes the location of each of the object on each of the images by defining the coordinates of the top left and bottom right points of it.

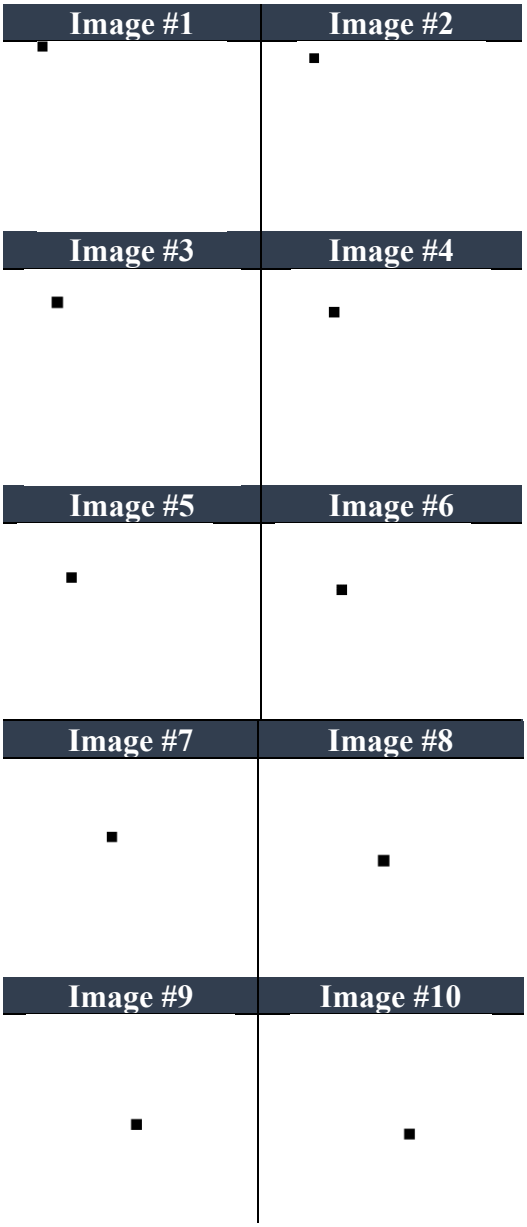
The data has the following appearance, with the white and block being the generated image and the red rectangle being the representation of the generated location coordinates.



The created csv file has the following structure>

	A	B	C	D	E
1		0	1	2	3
2	0	222	146	285	199
3	1	267	64	357	124
4	2	164	170	229	262
5	3	51	47	149	105
6	4	314	60	360	158

For the sequential set the generated dataset has the following appearance:



5. Performance measurements

For the evaluation of the performance of the project the conventional accuracy and loss measurements have been implemented.

Concretely, for the loss the Mean Squared Error loss has been used. Even if the model is of regression type the accuracy values have also been used to analyze the performance.

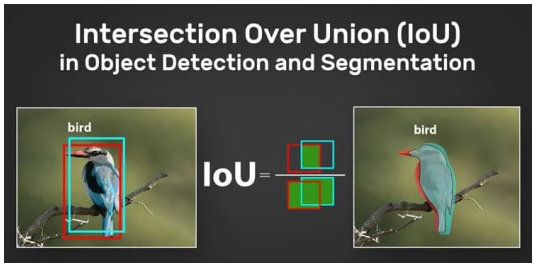
Moreover, to check the accuracy of our model when predicting the object's next position another indicator was necessary. The points of the location can be far from the location there should, but the area covered may still be good enough.

To do so, a common indicator in computer vision has been used: 'Intersection Over Union' or 'IoU'.

5.1. Intersection Over Union (IoU)

This method roughly measures the overlap existing between two boxes. IoU assesses the overlap of the Ground Truth and Prediction region in the context of object identification and segmentation.

That is the quotient between the area of the overlapping and the area of the union as shown in the figure below.



6. Results

After developing the different models, based on the step-by-step approach described at the beginning of this document it can be said that the obtained results are good enough to have a good approximation to the potential of the techniques, but also to the reality and complexity of the matter in hand.

The next subsections will show the obtained results on each of the developed models.

6.1. Object Detection

After several configurations, the finally implemented model presents the following network architecture:

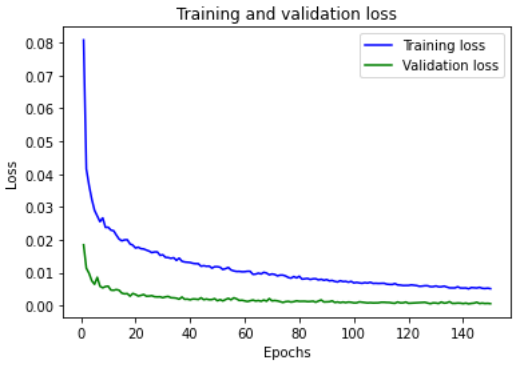
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 400)               64000400

 dense_1 (Dense)             (None, 200)               80200

 dense_2 (Dense)             (None, 100)               20100

 dense_3 (Dense)             (None, 50)                5050

 dropout (Dropout)           (None, 50)                0

 dense_4 (Dense)             (None, 4)                 204

=================================================================
Total params: 64,105,954
Trainable params: 64,105,954
Non-trainable params: 0
_________________________________________________________________
```

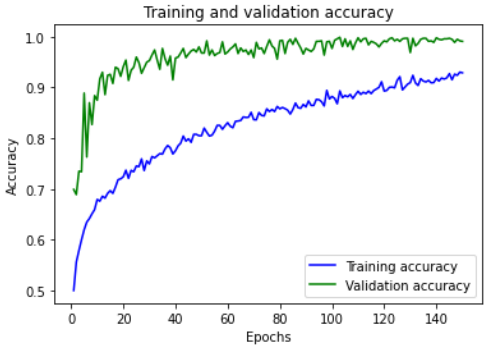
The network is built with 4 fully connected layers, first with a set of 400 units, second with 200, third with 100 and the fourth one being of 50 units. This is completed with a dropout and a final output layer that will show the 4 estimated values as described previously.

The given network has been also represented graphically and can be seen in the following figure:

dense_input	input:	[(None, 160000)]
InputLayer	output:	[(None, 160000)]
dense	input:	(None, 160000)
Dense	output:	(None, 400)
dense_1	input:	(None, 400)
Dense	output:	(None, 200)
dense_2	input:	(None, 200)
Dense	output:	(None, 100)
dense_3	input:	(None, 100)
Dense	output:	(None, 50)
dropout	input:	(None, 50)
Dropout	output:	(None, 50)
dense_4	input:	(None, 50)
Dense	output:	(None, 4)

In terms of performance, the execution of the object detection model has shown the following results:



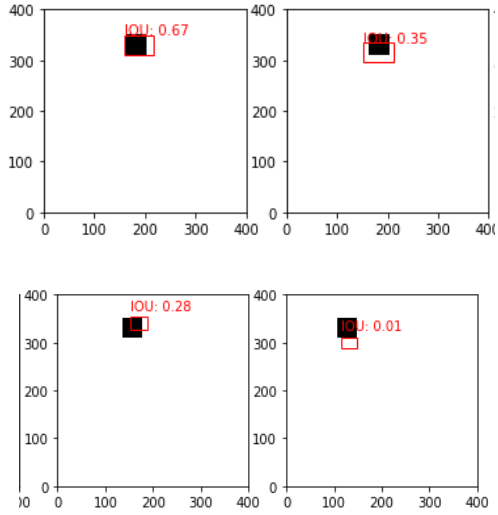


Additionally, the model shows the following IoU level:

```
Actual accuracy level of the system:  0.5040869654983103
```

Which means that the model always estimates where half of the object is that is willing to detect.

This is traduced to the following results:



We can see that the model can efficiently detect the objects and bound them.

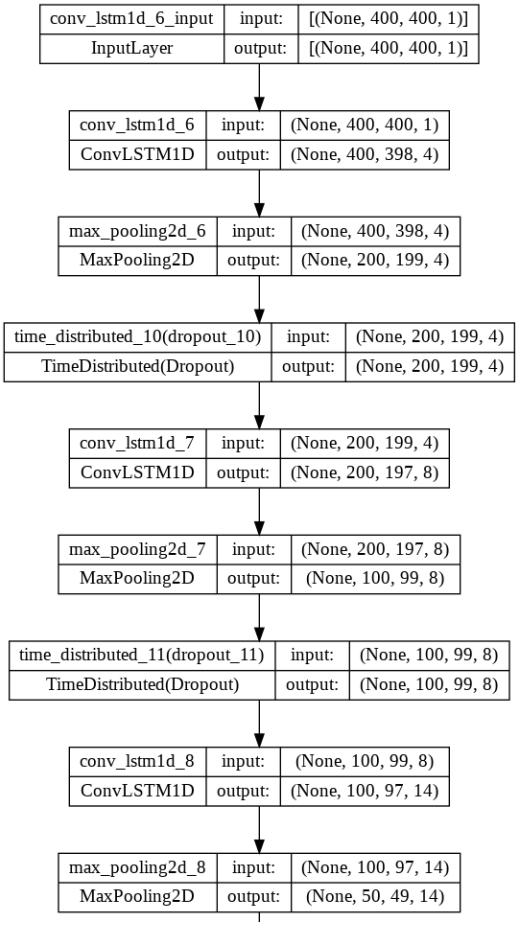
6.2. Object Tracking

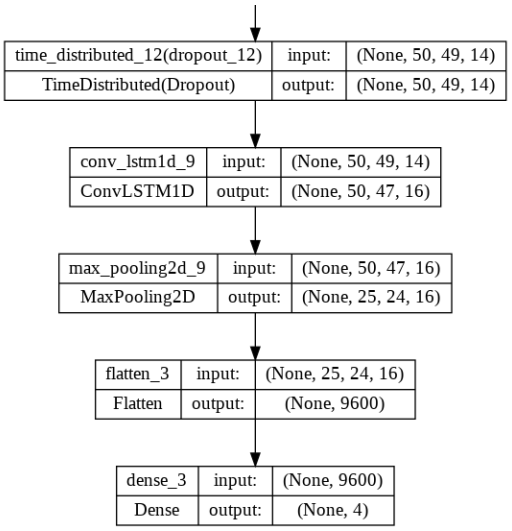
Related to the second approach the network has considerably increased the complexity by the addition of several layers of ConvLSTM type as mentioned in previous section to try to get the best results.

The network presents the following architecture:

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv_lstm1d_6 (ConvLSTM1D)  (None, 400, 398, 4)       256

 max_pooling2d_6 (MaxPooling  (None, 200, 199, 4)      0
 2D)

 time_distributed_10 (TimeDi  (None, 200, 199, 4)      0
 stributed)

 conv_lstm1d_7 (ConvLSTM1D)  (None, 200, 197, 8)       1184

 max_pooling2d_7 (MaxPooling  (None, 100, 99, 8)       0
 2D)

 time_distributed_11 (TimeDi  (None, 100, 99, 8)       0
 stributed)

 conv_lstm1d_8 (ConvLSTM1D)  (None, 100, 97, 14)       3752

 max_pooling2d_8 (MaxPooling  (None, 50, 49, 14)       0
 2D)

 time_distributed_12 (TimeDi  (None, 50, 49, 14)       0
 stributed)

 conv_lstm1d_9 (ConvLSTM1D)  (None, 50, 47, 16)        5824

 max_pooling2d_9 (MaxPooling  (None, 25, 24, 16)       0
 2D)

 flatten_3 (Flatten)         (None, 9600)              0

 dense_3 (Dense)             (None, 4)                 38404

=================================================================
Total params: 49,420
Trainable params: 49,420
Non-trainable params: 0
_________________________________________________________________
```

Which can be represented graphically with the following schema:

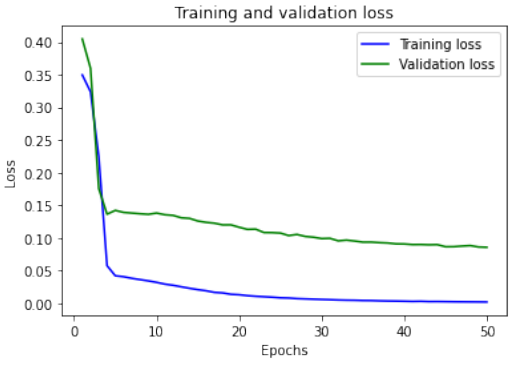




For this case, the obtained results are significantly worse as the model is not able to correctly predict.

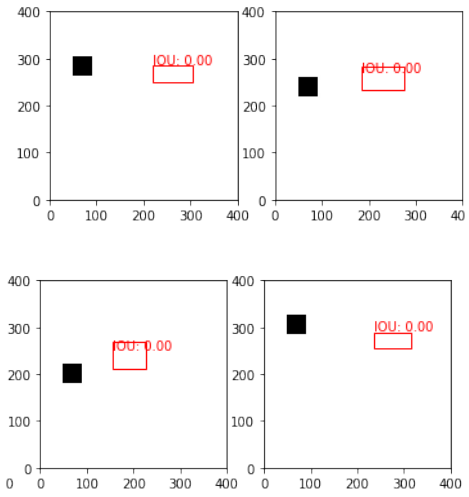
This is a result of very low computational capacity and training time. The model can only be run with very few data. Additionally, the hyperparameters must be set to not optimal values as the environment has no resources to run it otherwise.

The loss and accuracy are the following ones:



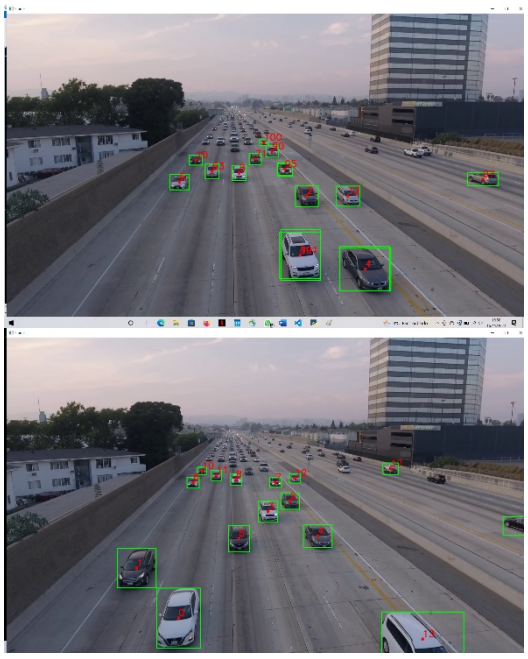
The loss shows how a better trained model would be able to reach much better results as it has a pretty good tendency.

For this particular case, the predicted data do not show the expected results at all:



6.3. Third Party Libraries

As mentioned before we have made use of the OpenCV DNN module, and executed it on a video which depicts several cars on the Los Angeles highway. When we execute it, we can see from the figure below, that several cars are detected and a bounding box for each of them appears. Moreover, the software also indicates which designated car appears and which are the coordinates of the prediction. That is that since the execution of the video is slow, it appears on the command prompt frame by frame which are the cars detected.



```
Tracking objects
{2: (1190, 844), 8: (797, 757), 10: (553, 618),
 (643, 449), 191: (935, 447), 193: (320, 958),
```

```
Tracking objects
{2: (1145, 779), 8: (788, 701), 10: (569, 589)
754, 444), 184: (395, 841), 185: (1610, 517),
```

7. Conclusions

To conclude, after a thorough examination of the field of computer vision, we have seen that there is multiple pre-trained software to execute the task we were trying to accomplish.

Moreover, we have seen in firsthand, the complexity and the greatness of the computational cost required for this kind of projects. Not in vain, multiple companies such as Tesla has invested a great amount of money and resources to excel in the computer-vision field.

However, and since it is not the point of this project, we have created everything from scratch with a dataset of our own and with a neural network of our own. We have studied which models and which neural networks work better for object recognition (RNN) and have made use of them.

Finally, and judging the results obtained, we can state that the results are considerably good, since the principal objective of the paper has been fulfilled, the detection and tracking of objects using recurrent neural networks.

8. References

[1] “OpenCV's DNN module and Deep Learning (a definitive guide),” *LearnOpenCV*, 18-Oct-2021. [Online]. Available: https://learnopencv.com/deep-learning-

with-opencvs-dnn-module-a-definitive-guide/. [Accessed: 13-Nov-2022].

[2] “Real-time recurrent regression networks for visual tracking of generic objects,” *IEEE Xplore*. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8253805. [Accessed: 13-Nov-2022].

[3] “Recurrent neural networks (RNN) with Keras : Tensorflow Core,” *TensorFlow*. [Online]. Available: https://www.tensorflow.org/guide/keras/rnn. [Accessed: 13-Nov-2022].

[4] J. Brownlee, “CNN long short-term memory networks,” *MachineLearningMastery.com*, 14-Aug-2019. [Online]. Available: https://machinelearningmastery.com/cnn-long-short-term-memory-networks/. [Accessed: 13-Nov-2022].

[5] “Object detection with neural networks - towards Data Science.” [Online]. Available: https://towardsdatascience.com/object-detection-with-neural-networks-a4e2c46b4491. [Accessed: 14-Nov-2022].