# AeronauticalObject Recognitionusing Neural Network Algorithms

Gjorgji Markovski[1[]], LeonidDjinevski[1[]] ,Slavcho Chungurski[1[]] andToni Stojanovski[2[]]

[1]FON University, bul. KjiroGligorov 5, 1000 Skopje, Macedonia
gorgi.markovski@fon.mk, leonid.djinevski@fon.edu.mk,
slavcho.chungurski@fon.edu.mk
[2]Evolve IS, Melbourne, Australia
stojanovski.toni@gmail.com

**Abstract.** Aeronautical navigation datais discussed and possible issues regarding obtaining the source information is presented. AI algorithms for object recognition as presented in order to provide an automated approach for extracting publicly available information that is provided in image formats. The results prove a large accuracy for recognizing aeronautical data.

**Keywords:**AI, Neural Networks, Object Recognition, YOLO, SSD.

## 1 Introduction

The field of object recognition is practicing quite challenging tasks in computer vision. The term object recognition describes a set of computer vision processes that are closely related and introduce identifying and classifying objects in images.

Aeronautical navigation data sources information that could be finds in graphical format. Thus, having the ability to recognize aeronautical objects and text can be a driver for automating generation of data on an acceptable accurate level. The goal of this paper is to show that aeronautical objects can be recognized, and its context extracted from publicly available information using object recognition algorithms.

### 1.1 Paper Organization

This paper is consisted of four sections. After the Introduction, Aviation data issues are discussed in Section 2.To better understand the problem being developed, analytical method is used. Thus, the parts of computer vision, artificial intelligence, data science, object recognition, machine learning and in-depth learning are utilized in Section 3 of this paper. The obtained results presented in Section 3 are concluded in the last Section 4.

## 2 Aeronautical Data Issues

One of the broadest sources of aeronautical data is the ARINC 424 aeronautical navigation database [1]. There is a handful of certified navigation data providers around the world like Jeppesen [2], Navblue[3], Lido[4] that hold certificates from Civil Aviation Agencies. Although, the data is obtained from authoritative sources: such as the US DoD, FAA, and foreign Aeronautical Information Publications (AIP) which are mostly publicly available, the full ARING 424 database tend to be quite expensive. These sources do not always contain data that can be interfaced by third party applications, because are sometimes packaged as pdfs and images. Therefore, having a standardized database that is guaranteedby the aeronautical database providers for the data accuracy can justify the database price.

Another issue is the consistency of keeping the data UpToDate. The data is publishedin Aeronautical Information Regulation and Control (AIRAC) [5] cycles which are a period of 28 days.

## 3 Implementation

### 3.1 Methodology

To implement the needs of the application-practical part for the Yolo algorithm, the Linux Ubuntu operating system was used, which was launched on a virtual machine with the help of Oracle VM Virtualbox Manager, where the required desktop was set, while for the SSD algorithm Google Colab was used.

For the purpose of doing an implementation part, images from Google Maps are used to recognize and write the text contained in the images given as input parameters (images) as well as images from Google Street View from world famous airports (for example London airport), traffic signs, etc., where the text of the images is recognized.

A custom database is used, which consists of 470 images. The image tool https://imglab.in/ was used to label and get the text coordinates. The text recognition is done with the help of Tesseract and the resulting images are recognized, together with the coordinates are written in an initially empty .txt document, while the images containing text are stored in the "Crop" folder. As a final stage of the application part is made between the two algorithms, where the speed of execution and the accuracy of the obtained results are compared. According to is made with the help of Medium Central Precision Object Detection (mAP). In the section for YOLO, MobileNetV2 is used and for the image labeling process, two folders are used, where in one of the coordinate members and in the other text (ground true folder). MobileNetV1 is used in the SSD section and XML is used for the image labeling process which contains the coordinates and the text and serves as the input (input) for creating CSV documents.

### 3.2 Helper apps and libraries

In order to implement the object recognition algorithms, additional apps and libraries were used in order to facilitate the data processing.

Some of these are: NumPy which is a Python general-purpose array processing package that Provides a multi-dimensional array of high-performance objects and tools for processing these arrays. Pandas for conversion and manipulation of data. Visual progress presentation with tqdm. Plotting charts with Matplotlib.pyplot. Python – tesseract for OCR (Optical Character Recognition). Object detection with OpenCV. A console user friendly interface for facilitating arguments with Argparse.TensorFlowas Neural Network library to apply AI algorithms. Code reusability with Utils. Keras for rapid building of Neural Networks.

### 3.3 YOLO

The Jupyter Notebook was used to execute our code. First, we imported the required libraries into Python such as Keras, matplotlib, scipy, os, cv2, and so on. Next, we declared variables to load the data and define the path of that data. Next, we initialize the image variables, the grid's height and width, and height and width of the image. Next, images in the database consist of different sizes and noise. For the above reason, we change the size of the images to dimensions of 512 x 512. The basic coordinates of the truth are processed to form a matrix with dimensions (grid height, grid width, 1, 5). The coordinates of the images are saved as NumPy arrays in their corresponding paths. Utils.py defines all the functions needed to further execute the code. The purpose of this document is to avoid duplication of code and to obtain greater visibility in the operation. The same document defines functions such as IoU, non_max, decode_to_boxes, etc.

### 3.4 Results

The obtained results are discussed and depicted in the figures in this section. The figures are consisted of two parts (left and right)which are representing the YOLO and SSD algorithms respectively. Additional tabular representation of the obtained results is given for each figure.

The SSD algorithm was executed on Google Colab on the GPU, while the YOLO algorithm on the Linux virtual machine CPU on a laptop machine.

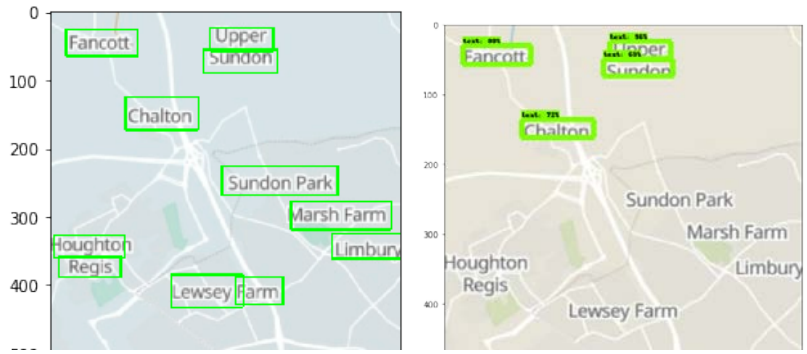


**Fig. 1.**Detection of text in sample 1.

**Table 1.**Performance table for processing sample 1.

| YOLO | | | SSD | | |
|---|---|---|---|---|---|
| Bounding Boxes (x,y,w,h) | Scores | Tesseract | Bounding Boxes (x,y,w,h) | Scores | Tesseract |
| 234 24 326 58 | 0.99 | iUpp_er | 236 24 324 49 | 0.97 | Upper |
| 23 26 128 65 | 0.97 | Fancdtt | 26 30 124 56 | 0.80 | Fancott. |
| 110 125 216 173 | 0.94 | Chalton | 110 136 214 161 | 0.72 | Chalton |
| 224 56 332 90 | 0.98 | Sundon | 227 49 327 73 | 0.68 | CGinnan |
| 13 357 103 389 | 0.99 | Regis | | | |
| 177 385 282 432 | 0.99 | Lewsey F | | | |
| 251 226 420 269 | 0.98 | Sundon Park | | | |
| 271 388 340 428 | 0.97 | Fa rm | | | |
| 412 324 519 361 | 0.96 | Limbury | | | |
| 352 277 498 319 | 0.94 | Marsh Farm | | | |
| 5 327 108 360 | 0.86 | ioughtm | | | |

Processing Time: 42.546 SecondsProcessing Time: 2.909 Seconds

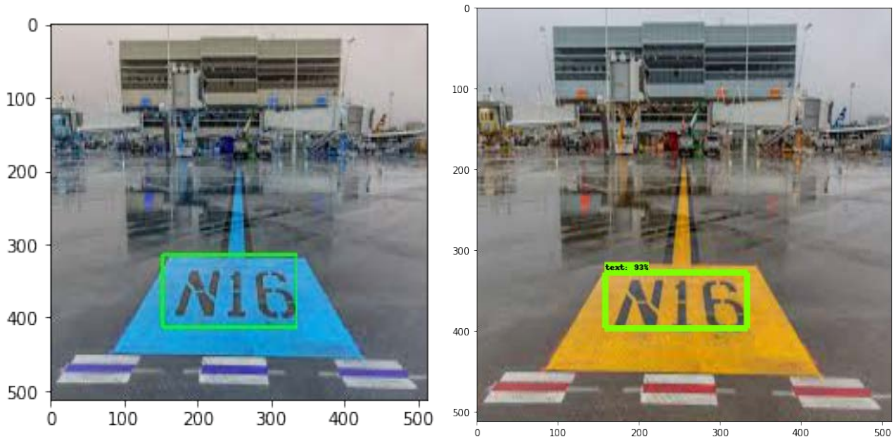


**Fig. 2.** Detection of text in sample 2.

**Table 2.**Performance table for processing sample 2.

| YOLO | | | SSD | | |
|---|---|---|---|---|---|
| Bounding Boxes (x,y,w,h) | Scores | Tesseract | Bounding Boxes (x,y,w,h) | Scores | Tesseract |
| 152 314 334 413 | 0.93 | 7x716 | 158 328 334 396 | 0.93 | AUG |
| Processing Time: 7.874 Seconds | | | Processing Time: 0.959 Seconds | | |

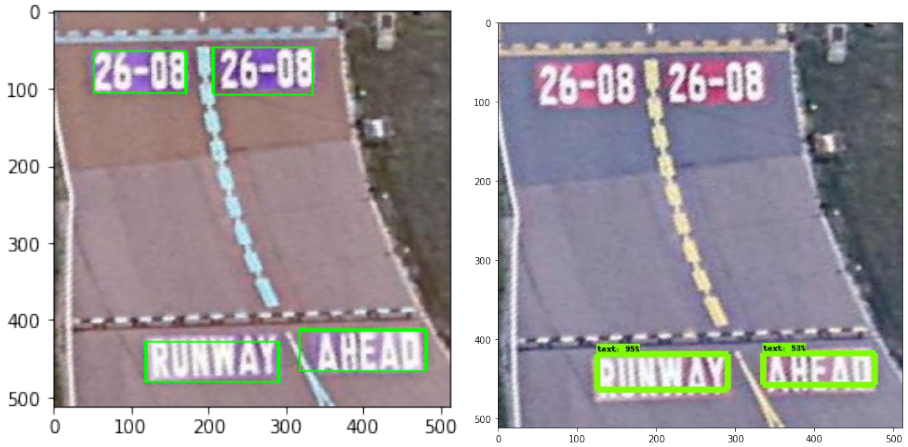


**Fig. 3.** Detection of text in sample 3.

**Table 3.**Performance table for processing sample 3.

| YOLO | SSD |
|---|---|

| Bounding Boxes (x,y,w,h) | Scores | Tesseract | Bounding Boxes (x,y,w,h) | Scores | Tesseract |
|---|---|---|---|---|---|
| 119 428 291 480 | 0.97 | | 125 419 291 463 | 0.94 | ohh in |
| 318 413 481 465 | 0.58 | | 335 418 477 457 | 0.52 | a |
| 53 52 171 107 | 0.83 | 26-08 | | | |
| 206 47 335 108 | 0.79 | 26-88 | | | |

Processing Time: 18.860SecondsProcessing Time:1.723 Seconds

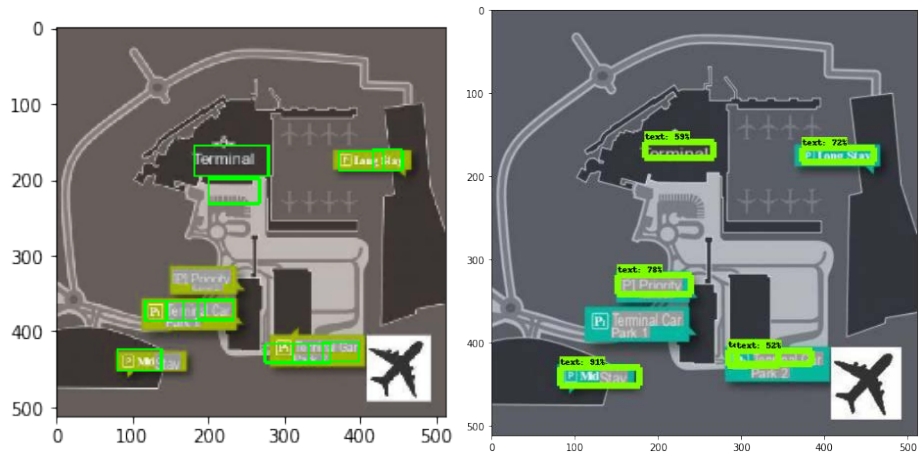


**Fig. 4.** Detection of text in sample 4.

**Table 4.**Performance table for processing sample 4.

| YOLO | | | SSD | | |
|---|---|---|---|---|---|
| Bounding Boxes (x,y,w,h) | Scores | Tesseract | Bounding Boxes (x,y,w,h) | Scores | Tesseract |
| 80 423 138 452 | 0.85 | Elli | 82 430 178 451 | 0.91 | ein ae |
| 200 199 267 232 | 0.58 | | 151 319 240 341 | 0.78 | (isd adda |
| 416 161 455 188 | 0.59 | as": | 373 166 460 182 | 0.72 | 'P) Tone Stay |
| 310 417 360 441 | 0.99 | или | 184 159 267 177 | 0.59 | PL TOY |
| 145 360 196 386 | 0.99 | | 284 409 345 427 | 0.59 | cee, |
| 186 359 234 385 | 0.99 | um | 294 410 385 425 | 0.52 | B.) Tannin-al GC: |
| 371 163 430 189 | 0.99 | Шилд: | | | |
| 182 156 279 196 | 0.97 | Terminal | | | |
| 350 414 398 440 | 0.96 | | | | |
| 117 360 168 386 | 0.95 | | | | |

274 417 331 441 0.62
200 199 267 232 0.58

Processing Time: 45.605SecondsProcessing Time:4.845 Seconds

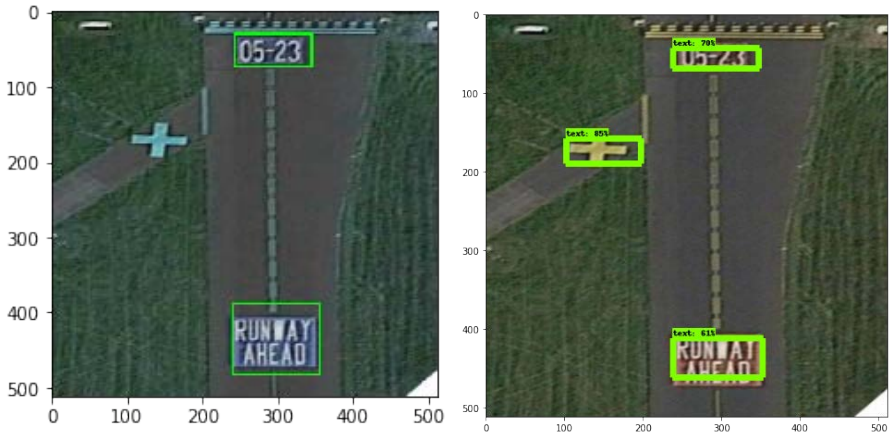


**Fig. 5.** Detection of text in sample 5.

**Table 5.**Performance table for processing sample 5.

| YOLO | | | SSD | | |
|---|---|---|---|---|---|
| Bounding Boxes (x,y,w,h) | Scores | Tesseract | Bounding Boxes (x,y,w,h) | Scores | Tesseract |
| 243 30 345 74 | 0.95 | 05-23 | 238 43 348 68 | 0.70 | UD-Z5 |
| 241 388 355 482 | 0.92 | ruin AHEAD | 238 412 353 461 | 0.61 | RUMEAT |
| | | | 102 158 198 189 | 0.84 | = |

Processing Time: 9.975SecondsProcessing Time:2.280Seconds

## 4 Conclusion

Our presented work is a case-study of the two models of neural network object detection neural network (YOLO and SSD) for the problem of localization in terrain images that involves many variables. The underling It should be noted that the problem under consideration is a greater challenge than the classical detection of "text in the wild", as there is no pre-defined texture pattern.

Since is hard to obtain tagged images for aeronautical data, we setup a database ourselves to conduct the detection. The comparisons obtained acceptable results for different sizes of panels, partial panel occlusion and complexity, perspective of images, conditions of lighting. The main power of the SSD approach was FP cases elimination that are preferred for implementations related to analyzing panels.

With the YOLO approach, we obtained better average results for detection because more TP panels were localized, proving higher accuracy relative to SSDs (relative to the corresponding basic truths of the images that we tested).

Our work involves comparisons with semantic segmentation networks. This is performed on similar problems, however with the same evaluation metric. We concluded that the accuracy that we obtained is similar. The images used in the experiment for testing the YOLO and SSD models are images on which the model has not been previously trained, which shows that the model works on random (random) images and thus proves its great application. Existing models can also be used to detect other types of objects, of course if we train them in pictures for other purposes.

Recognizing text in natural images is in direct relation to image quality. Thus higher the image quality that is being processed, the more likely it is to detect text. In the comparison and analysis that was done in this paper, the YOLO algorithm for object recognition resulted in more detected text objects and greater accuracy (score). It also shows the MAP score of 75% versus 38.95% accuracy of the SSD. Processing speed is better than the SSD algorithm, mostly due to the execution of a different environment.

## References


1. Airlines Electronic Engineering Committee, ARINC 424 Navigation System Data Base Standard, https://www.aviation-ia.com/activities/aeec.
2. Jeppesen. A Boeing Company, https://ww2.jeppesen.com/.
3. NavBlue, https://www.navblue.aero/product/navigation-plus/.
4. Lido/SkyData, Lufthansa Systems, https://www.lhsystems.com/tags/arinc-424.
5. Aeronautical Information Regulation and Control (AIRAC), ICAO, https://www.icao.int/airnavigation/information-management/Pages/AIRAC.aspx.