

הטכניון - מכון טכנולוגי לישראל

הפקולטה להנדסת חשמל ע"ש אנדרו וארנה ויטרבי

המעבדה לראייה ומדעי התמונה

**פרויקט ב'**

אימון רשת נוירונים לזיהוי רחפנים בסביבה אורבנית, הפעלת הרשת על כרטיסNVIDIA Jetson TX2 והתקנת המערכת על רחפן סיור

Training a neural network for detecting drones in an urban environment, integrate the network on an NVIDIA Jetson TX2 card and installing the system on a patrol drone

**מאת:**

אביב פסקרו Aviv Paskaro

סתיו יגר Stav Yeger

**מנחה:**

איליה צנוב Ilya Tcenov

**סמסטר רישום:**

חורף תש"פ

**תאריך הגשה:**

אוקטובר, 2020

# תוכן עניינים

[תוכן עניינים 1](#_Toc53293936)

[תקציר 2](#_Toc53293937)

[הפרויקט המקדים 4](#_Toc53293938)

[בחירת הרשת 5](#_Toc53293939)

[יצירת מאגר המידע ותהליך האימון 7](#_Toc53293940)

[אינטגרציה עם הכרטיס ותוצאות הרשת 15](#_Toc53293941)

[מדריך הורדת והפעלת הרשת 17](#_Toc53293942)

[סיכום ומסקנות 19](#_Toc53293943)

[בבליוגרפיה 20](#_Toc53293944)

[תודות 20](#_Toc53293945)

# תקציר

פרויקט זה הוא חלק בפיתוח מערכת לאיתור, עקיבה ויירוט רחפנים עוינים בסביבה עירונית. המערכת מבצעת את הניטור והסריקה של השטח על ידי רחפני סיור שמטרתם לצלם את הקרקע. את הצילומים האלו יש לעבד, והאפשרויות לכך הן עיבוד בעמדה קרקעית או עיבוד על גבי הרחפן. מכיוון שהמידע הגולמי הוא בצורת תמונה בעלת נפח איחסון גדול, זמן השידור המידע יהיה ארוך מאוד. אך, גם ביצוע עיבוד קלאסי על כרטיס המצורף לרחפן אינו יעמוד בדרישות הזמנים לעיבוד בזמן אמת (לעמוד בקצב עיבוד של לפחות 2 תמונות לשנייה). לכן, לצורך פתרון בעיה זו נבחר לאמן רשת נוירונים שתלמד לזהות את רחפנים בסביבה זו. הרשת תבצע את הזיהוי במהירות רבה, ותספק את מיקום הרחפן באמצעות קורדינטות שקל לשדר אל תחנת הקרקע במהירות רבה. בפרויקט מקדים יצרנו מאגר מידע גדול ומגוון, שיאפשר לאמן את הרשת אופייני טיסה שונים של רחפנים הטסים בסביבה עירונית. בפרויקט הנוכחי, כוח המיחשוב של המערכת הוא דל והמערכת בעלת הספק מוגבל, ולכן בחירת הרשת צריכה להיות בהתאם. בחרנו להשתמש ברשת מסוג Single Shot Detector (SSD) המתאימה לעבודה על מערכות דלות הספק, ומספקת ביצועים טובים בהיבט הזיהוי וזמן העיבוד. אימנו את הרשת שנבחרה במספר גדלים שונים (רזולוצית כניסת הקלט לרשת), דבר המאפשר השגת תוצאות מיטביות על ידי בחירת הארכיטקטורה המביאה את הביצועים הטובים ביותר, הן בהיבט דיוק והן בהיבט זמן חישוב. כמובן, לרשת הגדולה יש יתרון בזיהוי על פני הקטנה, אך החיסרון הוא בזמן העיבוד. בנוסף, התאמנו פרמטרים רבים עד לקבלת אחוז גבוה של זיהוי רחפנים בסרטוני המבחן. לבסוף, ביצענו אינטגרציה של הרשת המאומנת לכרטיס NVIDIA Jetson TX2 שנבחר לשימוש בפרויקט זה להיות על הרחפן בזמן ההפעלה, ובנוסף איגדנו את הפלט המיוצר בצורה מסודרת לצורך תקשורת עם מרכז הבקרה.

**Abstract**

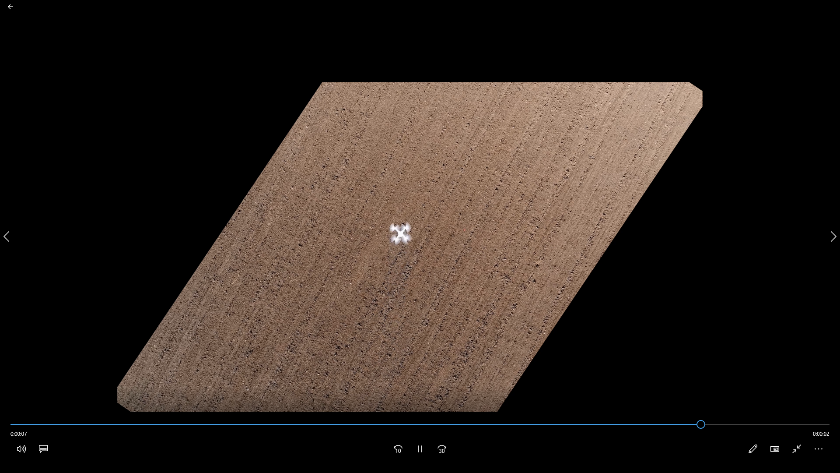
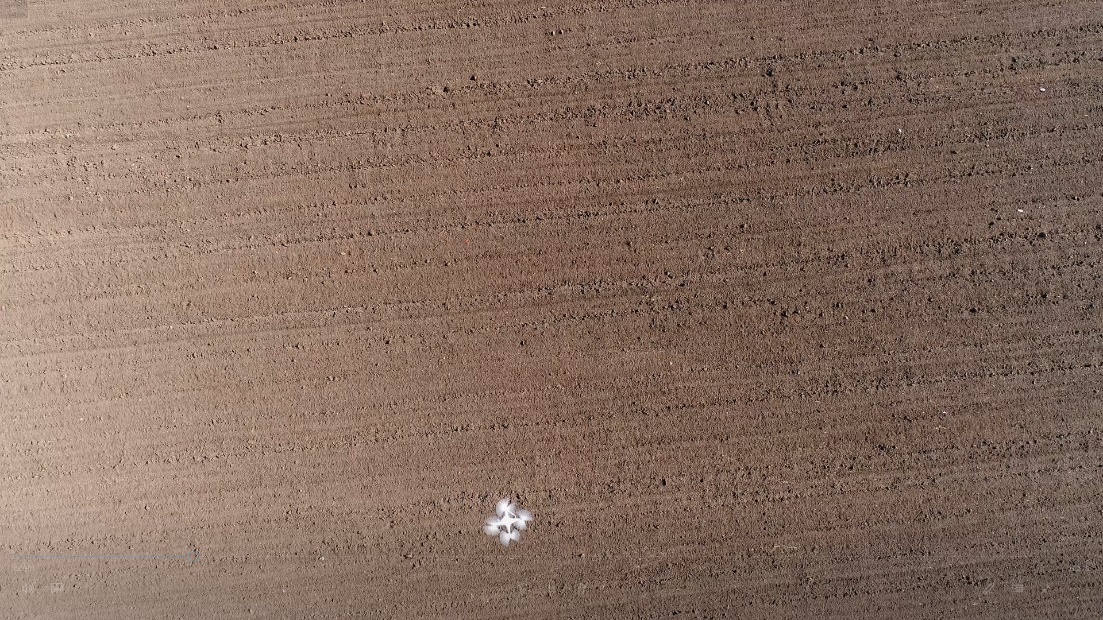
This project takes a step in developing a system for locating and tracking hostile drones, which penetrate an urban environment, in order to intercept them and thwart the threat. The system performs the monitoring and scanning of the area by patrol drones that aim to photograph the ground. The information that collected on the drones must be processed, and the options for this are to process at the ground station or on the drones. The raw information is in the form of an image and therefore the transmission time of the information is very long, but even performing image processing on a card attached to the drone does not meet the time requirements. To solve this problem, a neural network was chosen to learn to identify drones in this environment, the network will perform the identification very quickly and provide the location of the drones using coordinates, which will be easy to transmit to the ground very quickly. In the preliminary project, we created a large and diverse database, which will allow the network to train various flight characteristics of drones flying in an urban environment. In the current project, the computing power of the system is poor, and the system has limited power, so the choice of network should be accordingly. We chose to use a Single Shot Detector (SSD) network that is suitable for working on low power systems and provides good performance in terms of detection and processing time. We have trained the selected network in several different sizes (input size to the network), which allows for optimal results by choosing the architecture that brings the best performance, both in terms of accuracy and in terms of calculation time. Of course, the large network has an advantage in identification over the small one, but the disadvantage is in the processing time. In addition, we practiced many parameters until a high percentage of drone detection was obtained in the test videos. Finally, we integrated the trained network to the NVIDIA Jetson TX2 card selected for use in this project to be on the drone during operation, and in addition we bound the output produced in an orderly manner for communication with the control center.

# הפרויקט המקדים

בפרויקט המקדים יצרנו מאגר סרטונים המדמה תעופה של רחפנים בסביבה אורבנית. המאגר יוצר משילוב בין סרטוני רחפנים, הצולמו בשטח פתוח, לבין סרטונים של סביבה אורבנית הצולמו במספר מוקדים שבהם תנועה עירונית ניכרת. על ידי שימוש בכלים לעיבוד תמונה, סרטוני הרחפנים עברו הגדלות, סיבובים, הזזות, שינויי צבע ואף שילוב רחפנים בסצנה אחת. יתרה, השתמשנו בהגרלות רנדומליות, כל אלו על מנת לייצר מאגר גדול ומגוון שידמה, עד כמה שאפשר, סצנה פיסיקלית שבה רחפן נע בסביבה אורבנית.

ניתן למצוא את התוכן הפרויקט הקודם כולל התיעוד במאגר הבא:

<https://github.com/avivpaskaro/drones_project_data_augmentation>

נציג את רעיון הפרויקט הקודם בקווים מנחים:

4

3

1

2

הסרטון מאוחד לסצנה עירונית

הסרטון המקורי עובר אוגמנטציה

הסרטון עובר ניקוי מהרקע וישמש כמסכה לאיחוד בהמשך

סרטון שצולם בשטח פתוח



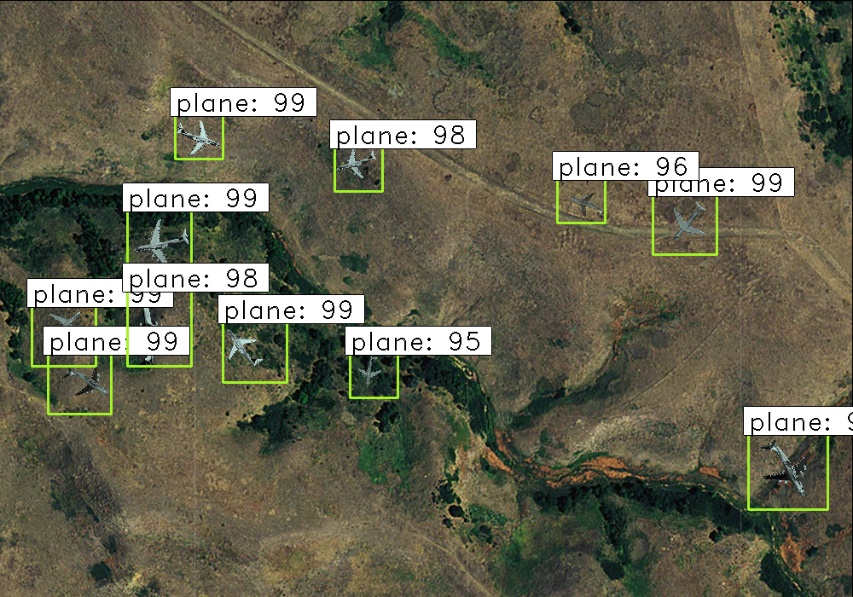
# בחירת הרשת

**השיקולים לבחירת הרשת:**

זיהוי ועקיבה הן לא אתגר חדש בשטח, ולו פתרונות רבים. עם זאת, רבים מפתרונות אלו מותאמים למערכות שלהן כוח ביצוע ומשאבים כמעט ולא מוגבלים, כגון חוות שרתים, מחשבי על ובאופן כללי מערכות מחשב בעלות אספקת חשמל קבועה. מנגד, קיימות מערכות רזות יותר שמטרתן להיות זולות וניידות, מה שיבוא על חשבון הביצועים. על מערכות שכאלו לא ניתן להפעיל כל תוכנה וכל רשת, ההספק מאוד מוגבל ומהירות הביצוע בהתאם.

מהצורך למערכת ניידת נאלצנו לבחור חומרה המיועדת לעבוד לאורך זמן עבודה מרווח. קיימות אפשרויות רבות לבחירת רשת שתבצע את המשימה. בין האפשרויות יש שתי גישות מקובלות, האחת היא לזהות את האובייקטים בתמונה בכל פריים ופריים בנפרד, וללא תלות בידע מקדים. השנייה היא להשתמש בידע על מיקום קודם ולבצע הערכה למיקום העתידי של האובייקט. כפי שניתן להסיק מכך, המערכות המשתמשות בזכרון הן כבדות יותר ובכך נותנות ביצועים נמוכים יותר על הכרטיסים הניידים. נציין עם כך שהמערכת הסופית שנדרשת בפרויקט זה צריכה לעמוד בקצב עיבוד של לפחות 2 תמונות לשנייה, מה שהתברר שלא קל להשגה על הכרטיסים הניידים.

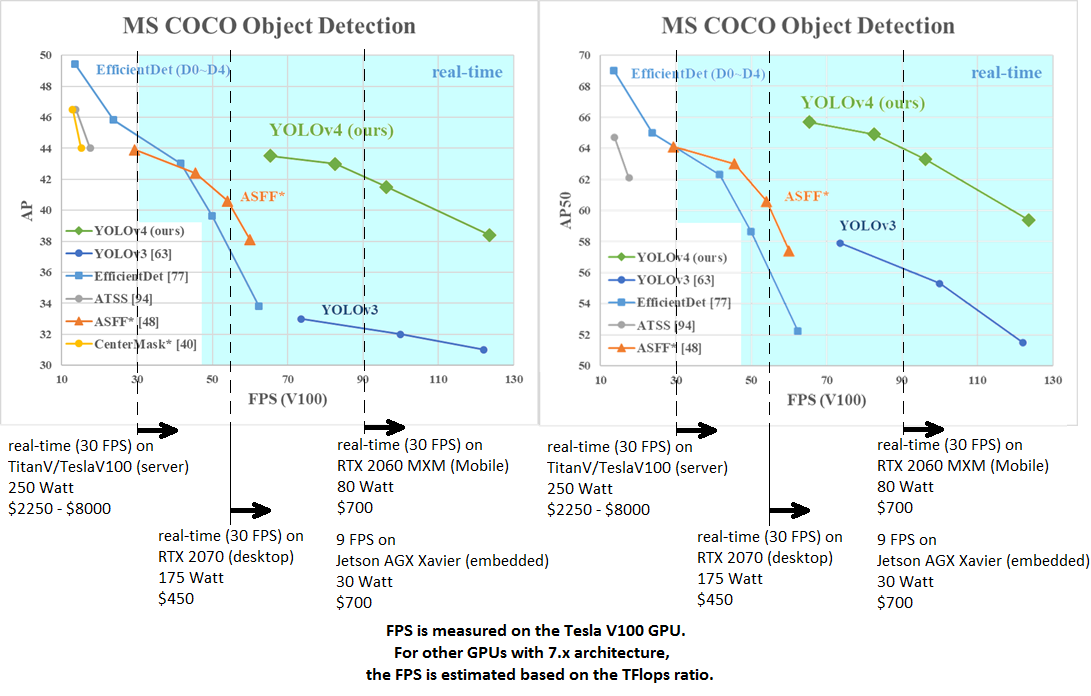
לאחר שביצענו סקר ספרות מקיף ובחנו ביצועים של רשתות שונות, התבררה לנו כי המועמדת הטובה ביותר היא SSD או Single Shot Detector. רשת זו מתבססת על הידע המיידי ולא מסתמכת על הידע הקודם. רשתות מסוג זה מחלקות את התמונה למסגרות רבות ומעריכות את הסיכוי להימצאות אובייקט בכל מסגרת ומסגרת. נראה דוגמה לתוצאה הסופית שבה רואים זיהוי אובייקט והערכה באחוזים למידת הודאות בסוג האוביקט, רשתות אלו הן הפתרון המושלם למערכת שלנו.



**הרשת שנבחרה:**

האופציות לבחירת רשת הן גדולות מאוד, ולכל רשת יש יתרונות וחסרונות. מפתחי הרשתות מתחרים על פיתוח רשת בעלת דיוק גבוה בזמן עיבוד מינימלי ככל הניתן. קיימים מאגרי תמונות שעליהם מקובל לבחון את הרשתות, הכוללים אובייקטים משלל קטגורויות שנפוצות בחיי היום יום. בין רשתות אלו ניתן למצוא שמות כמו: YOLO, RetinaNet, R-FCN, SSD, FPN,  ו- Faster R-CNN. מבין רשתות אלו הנפוצה ביותר היא ה-YOLO, רשת המשיגה את התוצאות המהירות ביותר, ובעלת דיוק שלא ניתן להתחרות בקצב עיבוד שכזה.

בדיאגרמות הבאות ניתן לראות ביצועים של רשתות שונות על מאגר זהה בשם COCO:



YOLOV4

YOLOV4

לפי תוצאות אלו, ובנוסף מהסיבה שYOLO היא רשת מאוד פופולרית וקיימות לה גרסאות ועדכונים רבים, בחרנו להשתמש בגירסה העדכנית שלה - YOLOv4. בשל הביצועים המהירים שלה היא מתאימה לעבוד על כרטיסים בעלי הספק נמוך ולספק את הסחורה. הגירסה העדכנית של YOLO היא הגירסה הרביעית שיכולה להגיע ל-AP של למעלה מ65%, ולבצע זאת בקצב עיבוד של כ55 פריימים לשנייה על כרטיס גרפי בעל ביצועים גבוהים. YOLO מבוססת על רשת קונבולוציה עצבית (CNN) כך שהיא מחלקת תמונה לאזורים, ועבור כל אזור מחשבת את ההסתברויות להמצאות אובייקטים. בכך, הרשת יכולה לזהות מספר אובייקטים במקביל ולעשות את במהירות רבה. אנו מסתמכים על כך שאם הביצועים של הרשת עבור GPU הם יותר טובים גם בקצב וגם בדיוק, אז עבור GPU חלש יותר נקבל תוצאות פחות טובות אך טובות משאר הרשתות. ביצועים של YOLO ניתן לראות בקישור הבא [YOLOv4](https://www.youtube.com/watch?v=1_SiUOYUoOI&feature=youtu.be).

# יצירת מאגר המידע ותהליך האימון

**המאגר:**

בפרויקט המקדים לא היה ידוע לנו סוג הרשת שבה נשתמש בהמשך, ולכן יצרנו את המאגר סרטונים בצורה מוכללת ככל הניתן. הסרטונים נוצרו באופן שבו ניתן לשלוט על אורך הסרטון הנוצר, על הרזולוציה ועל מספר העותקים הכולל. סוג הרשת שנבחרה הוא SSD, רשתות מסוג זה מתאמנות על תמונה בודדת ולא על סרטון, ולכן ביצענו עיבוד נוסף על מאגר הסרטונים שיצרנו. לקחנו את מאגר הסרטונים ופיצלנו אותו לאוסף תמונות, כך שלכל תמונה מצטרף קובץ טקסט המכיל את המידע על האוביקטים הנמצאים בה. המידע על האוביקט הוא בצורת מסגרת המקיפה את האוביקט, ובנוסף מספר המחלקה שלו. את הקורדינטות המתארות את מיקום וגודל המסגרת המקיפה את האוביקט לקחנו מתוך מפת האמת שנוצרה ביחד עם כל יחידת סרטון (מצורף כלי במאגר הקבצים לפרויקט זה שמטפל בהמרה הזו).

הפורמט של קובץ הטקסט הוא:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Height | Width | Y\_top | X\_left | Class Number |

**Class Number** – מספר המחלקה (למשל אם נרצה לאמן לזהות כלבים וחתולים אז כלבים יקבלו את המספר 0 וחתולים יקבלו את המספר 1).

**X\_left** - המרחק של המסגרת מהקצה השמאלי.

**Y\_top** - המרחק של המסגרת מהקצה העליו.

**Width** - רוחב המסגרת.

**Height** - אורך המסגרת.

Height

Width

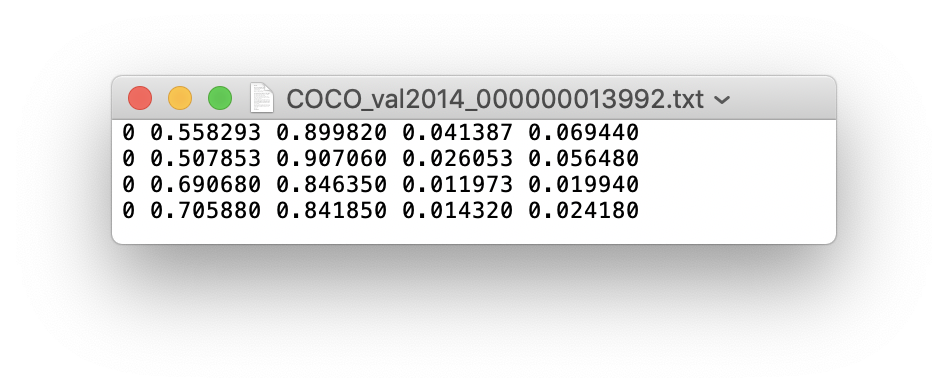
Object

X\_left

Y\_top

על מנת להימנע ממספרים גדולים, הרשת דורשת את היחס בלבד. כלומר, המספרים האלו הם בין 0 ל-1. למשל, אם אורך המסגרת הוא 100 פיקסלים ואורך של התמונה היא 1000 פיקסלים אז האורך שנשים הוא 0.1.

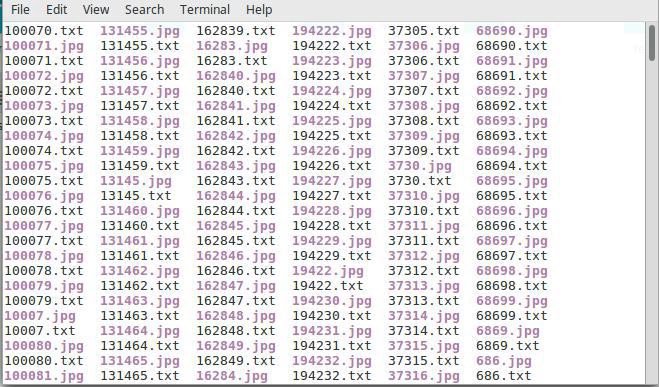
דוגמה לקובץ טקסט:



בדוגמה מתוארות מתוארים ארבעה אובייקטים מאותה המחלקה.

לצורך אימון הרשת לזיהוי הרחפנים יצרנו מאגר של כ150 אלף תמונות שבהן מספר רחפנים בתמונה בצבעים שונים, גדלים שונים ומיקומים שונים. את מאגר התמונות ריכזנו בתיקייה מרכזית, בנוסף עם קבצי הטקסט המתארים את האובייקטים. כל תמונה וקובץ טקסט תואם יקראו באותו השם למעט הסיומת.

לדוגמה:



**האימון:**

את תהליך האימון ניתן לעשות על פלטפורמות רבות כמו Windows או Linux, על מחשב אישי או בענן. במאגר של מפתחי הרשת קיימות הוראות לדרכי אימון שונות, וניתן למצוא באינטרנט דרכים רבות נוספות. נצרף את הקישור למאגר הרשת המקורי ולא נפרט את השלבים בפרטי פרטים כאן. הסיבה לכך היא שהמאגר מתעדכן לעיתים קרובות, ואיתו גם הוראות האימון. אך, בכל זאת, נסביר על השלבים באופן כללי.

תהליך האימון הרשת התבצע בשלבים הבאים:

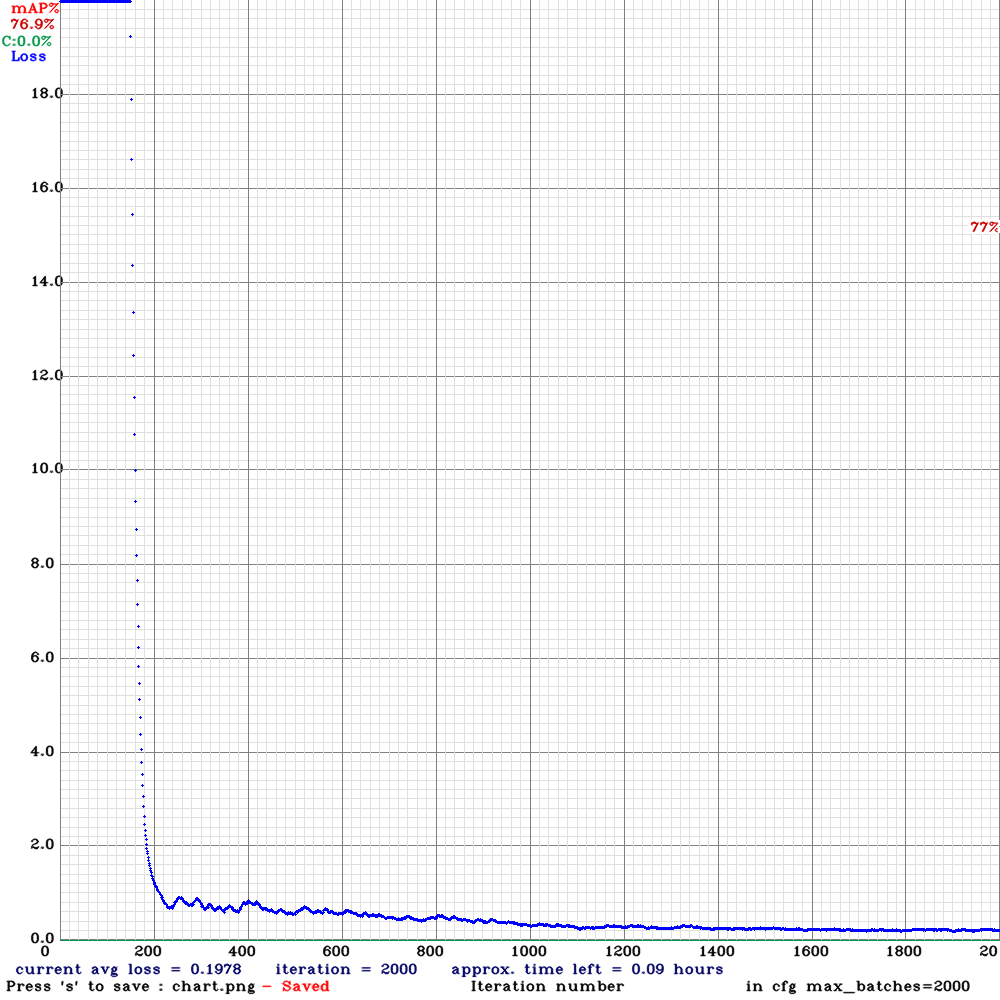
1. הורדת קבצי המקור מתוך המאגר [YOLOV4 GitHub](https://github.com/AlexeyAB/darknet)
2. ביצוע קומפילציה של הרשת בהתאם למערכת ההפעלה ולחומרה. לפני ביצוע הקומפילציה יש לגשת לקובץ ה-Makefile שבין הקבצים שהורדנו ולשנות את הפרמטרים המתאימים.
3. לאחר שהרשת קומפלה בהצלחה, יש להוריד את קובץ המכיל משקלים pre-trainedכדי לתת למערכת נקודת התחלה מתקדמת לאימון.
4. יש למקם את מאגר התמונות וקבצי הטקסט בתיקיה ייעודית.
5. יש להגדיר מספר קבצי קונפיגורציה, ביניהם כאלו שיכילו בתוכם את המאפיינים של הרשת מבחינת גדלים ומחלקות, ובנוסף יתארו את מיקומי התמונות לאימון.
6. התחלת האימון בעזרת פקודה הרצה מהטרמינל.

את האימון ביצענו בארבע קונפיגורציות שונות, בכל אחת שינינו את גודל הכניסה לרשת, דבר שמשפיע על קצב עיבוד הרשת והדיוק של הזיהוי. כל אימון הופסק בצורה אוטומטית, על ידי מספר חזרות שנקבע מראש לפי המלצת מפתחי הרשת. בכל אימון שכזה הרשת הגיע לדיוק של כמעט ומקסימלי על סט האימון. נציג את תוצאות האימון על ידי גרפים המתארים את התהליך. בכל גרף ניתן לראות את פונקצית השגיאה, וגם את חישוב mAP שהתבצע על סט הבוחן בסוף האימון.

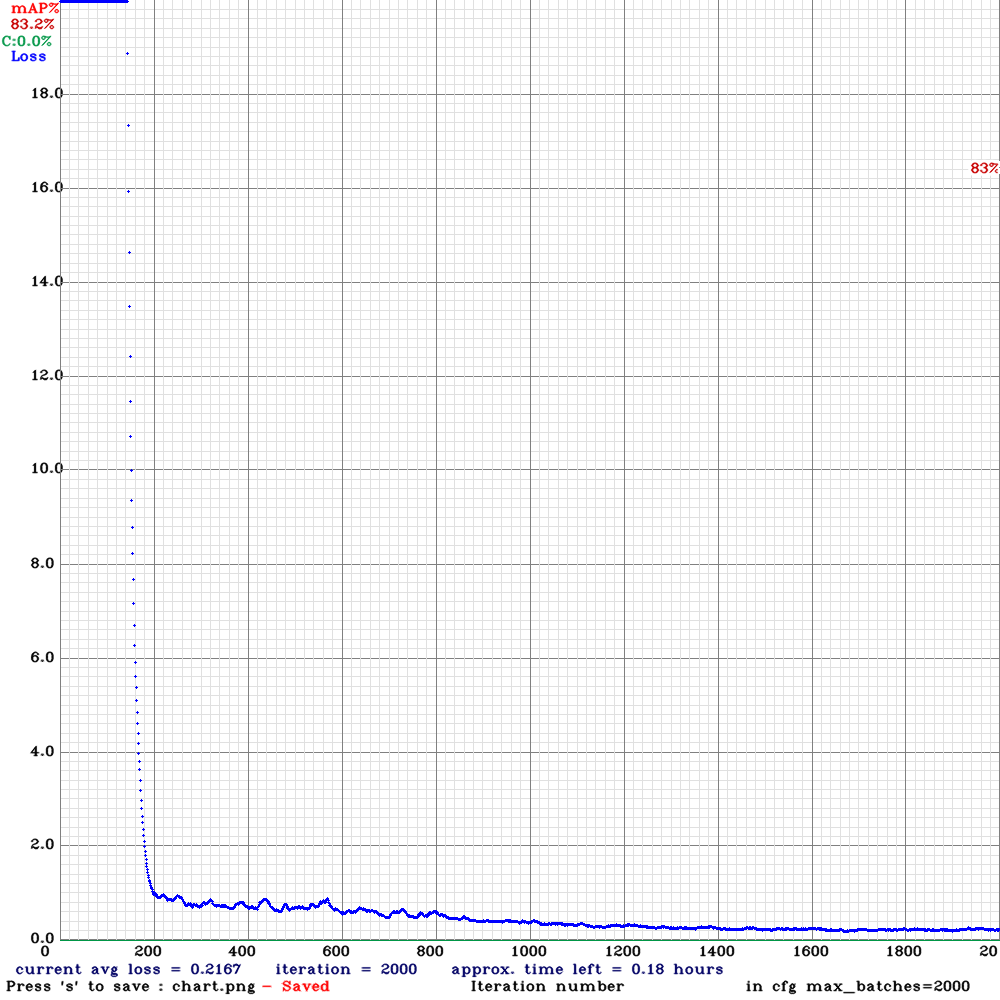
גרף אימון של רשת בגודל כניסה של 320X320:



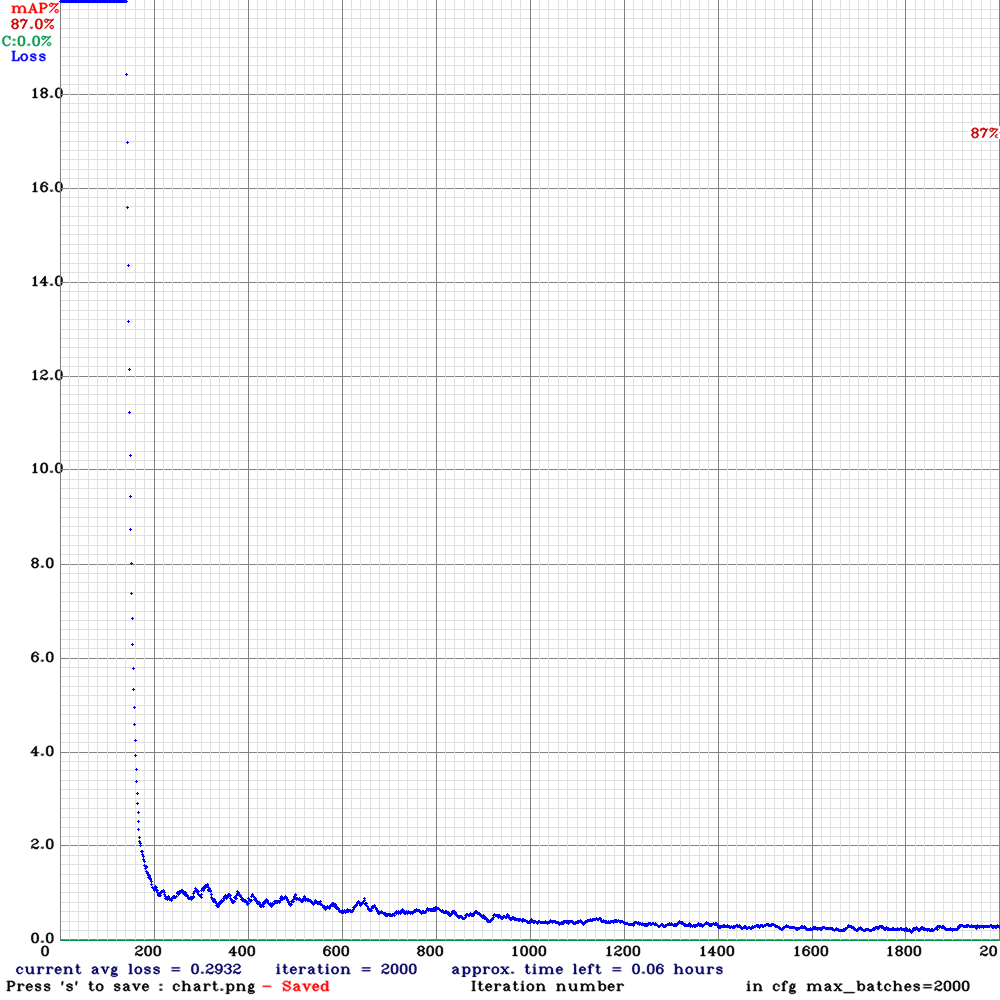
גרף אימון של רשת בגודל כניסה של 416X416:



גרף אימון של רשת בגודל כניסה של 512X512:



גרף אימון של רשת בגודל כניסה של 608X608:



התוצאות:

בכל הגרפים ניתן לראות התכנסות של גרף השגיאה למספרים נמוכים מאוד, עד כדי הבדלים בספרות שאחרי הנקודה העשרונית. הגדלים האלו נבחרו שרירותית וניתן לבחור גודל כניסה לרשת שהוא כפולה שלמה של 32. נציין שהדיוק בסוף האימון על ה- validation-setהיה קונסיסטנטי עם הדיוק על ה training-set שלא נראה בזמן האימון. נסכם את התוצאות של הדיוק הממוצע על גבי סט הבוחן בכל אימון שביצענו:

* 320X320 = 65% mAP
* mAP 416X416 = 77%
* 512X512 = 83% mAP
* 608X608 = 87% mAP

# אינטגרציה עם הכרטיס ותוצאות הרשת

הכרטיס שנבחר לשימוש בפרויקט זה הוא כרטיס Jetson TX2 של חברת NVIDIA. זהו כרטיס יעודי ליישומים מתחום הבינה המלאכותית, הוא בעל חומרה יעודית שמטרתה להאיץ תהליכי אימון והרצת רשתות נוירונים. על הכרטיס מותקנת מערכת ההפעלה של לינוקס ולכך התאמנו את הקוד של הרשת, ובנוסף קיימת אפשרות בכרטיס להרצה בקונפיגורציה(MAX N) שמפעילה את המעבדים בתדרים גבוהים על מנת לדמות overclocking. בפרויקט זה בחרנו להשתמש ברשת בעלת גודל הכניסה הגדול ביותר שאימנו,(608X608) אשר אחוזי הדיוק שלה הם הגבוהים ביותר. למרות הגודל של הרשת אשר מגדיל את זמני העיבוד, עמדנו בדרישות הזמן שהגדירה התחנה הקרקעית.

נציג את התוצאות של זיהוי רחפנים שבוצעו על סרטונים שצולמו ממצלמת רחפן בסביבה אורבנית. הסרטונים לא ערוכים ומדמים מקרי אמת שאליהם נערכנו. המצלמה הורכבה על רחפן בגובה סיור של 50 מטר, והרחפן העוין טס בגובה של 10 מטרים בלבד. הסרטונים האלו צולמו באור יום באזור עירוני עם הולכי רגל ורכבים נוסעים. לרשת יש יכולת לזהות כשלושה רחפנים במקביל, למרות שבסרטונים מופיע רק רחפן בודד (אילוצי תיפעול).

תוצאות (ניתן לצופת בסרטון המלא [כאן](https://drive.google.com/file/d/1zU43xv_nh4M0in1JhqWPMN7T5VTQlIMn/view?usp=sharing)):



# מדריך הורדת והפעלת הרשת

על מנת להפעיל את הרשת יש לעקוב אחרי הוראות ההורדה והקימפול שנמצאים בRepository המקורי (מצורף לינק בפרק קודם). בנוסף, סיפקנו את קובץ הMakefile שאיתו עבדנו על הכרטיס וגם את קבצי הקונפיגורציות של הרשת. את הקבצים ניתן למצוא במאגר הבא:

<https://github.com/avivpaskaro/drones_project_yolov4_nn.git>

מדריך להפעלת רשת YOLOV4 על JETSON TX2:

דרישות:



הורדת הרשת:

1. git clone https://github.com/AlexeyAB/darknet.git
2. git clone https://github.com/avivpaskaro/drones\_project\_yolov4\_nn.git
3. cp ./drones\_project\_yolov4\_nn/\* ./darknet/
4. Download weights into darknet folder-

**YOLO 320X320:** https://drive.google.com/file/d/110cYGgU5lA1Jdj1a1nuDyVbDc6iq\_BHY/view?usp=sharing

**YOLO 416X416:** https://drive.google.com/file/d/1nLzVdx2llWd4oBnEyvvSRtFFvYPjyhhL/view?usp=sharing

**YOLO 512X512:** https://drive.google.com/file/d/1COKM\_HfefGoE\_tUVnkpf4e5j6zCw0MRv/view?usp=sharing

**YOLO 608X608:** https://drive.google.com/file/d/1mww1NgBP\_dfmh2JeRACX8rIUIWnOdNVO/view?usp=sharing

1. Download test video into darknet folder for checking the net- **test2:** https://drive.google.com/file/d/1stfZl-zn6wuOLQRvATd4D3OhpLozGip6/view?usp=sharing

הפעלת הרשת:

1. cd darknet
2. make
3. chmod 777 darknet\_run
4. ./darknet\_run

פרמטרים לריצה:

שורת ההרצה נמצאת בתוך הקובץ darknet\_run. את הפרמטרים ניתן לשנות בתוך הקובץ לפני האפשרויות הבאות:

--input - takes webcam number or video input path (webcam=0 if empty)

--out\_filename - takes out video file name (not saved if empty)

--export\_logname - takes out log file name

--weights - takes yolo weights path

--dont\_show - hide window inference display

--config\_file - takes path to config file

--data\_file - takes path to data file

--thresh - takes value to remove detections with confidence below (thresh=0.25 if empty)

--capture\_frame\_width - define the camera frame width

--capture\_frame\_height - define the camera frame height

לדוגמה (ריצה):

python3 -u darknet\_video.py --data\_file obj.data --config\_file yolo608-obj.cfg --weights yolo608-obj\_best.weights --thresh 0.25 --dont\_show --input test2.mp4 --export\_logname results.txt --out\_filename out.mp4

# סיכום ומסקנות

פרויקט זה התמקד בבעיית הראייה הממוחשבת של זיהוי עצמים בתמונות דינאמיות. בעיה זו היא מורכבת ומלאת גישות והחשיפה לבעיה זו ושלל הגישות לפתרונה לימד אותנו רבות על אופי החשיבה הנדרש לבעיות הנדסיות שכאלו, ובפרט נתן לנו כלים להתמודדות עם בעיות דומות בעתיד עקב גנריות(כלליות) הבעיה – בפרויקט זה מצאנו פתרון עבור רחפנים אך כמובן שבעזרת אימוץ גישות שונות אפשר למצוא עצמים נוספים ולזהות תבניות אחרות.

קיימת פשרה (trade-off) בין ביצועי המערכת לבין ההספק וכוח העיבוד שנדרש לביצוע המשימה מה שהפך את הפרויקט לבעיית אופטימיזציה מעניינת ומאתגרת. בחרנו ברשת הלמידה לזיהוי אובייקטים YOLOv4 מכיוון שהגענו למסקנה שהיא מחד גיסא תביא לנו ביצועים טובים ומאידך גיסא תעמוד בדרישות הספק ויעילות חישובית שיאפשרו קצב גבוה מספיק שיאפשר הרצה בזמן אמת.

מבחירת הרשת ועד לתפקוד והצגת ביצועיה, עברנו תהליך ארוך שכלל מחקר על אופן אימון הרשת:

* שימוש נכון במאגר המידע שיצרנו וחלוקתו לקבוצות אימון ומבחן
* אופטימיזציות לשיפור ביצועים באימון רשת
* שימוש בסביבות עבודה מקובלות בתחום כמו OpenCV ועבודה על GPUs

בנוסף עסקנו בהטמעת הרשת לכרטיס שיישב על הרחפן, שכללה אינטגרציה עם רכיבי הכרטיס ובנוסף עסקנו במציאת נקודות איזון של פרמטרים ושימוש באופטימיזציות לשם קצב גבוה ומניעת התחממות יתר.

לפי התוצאות המוצגות, מטרתנו הכללית הושגה. נציין כי הפתרון שבחרנו הוא אחד מני רבים וניתן להגיע לפתרונות לבעיה בכל מיני דרכים וגישות. אך לדעתנו במסגרת הפרויקט והיקף העבודה, הפתרון בו בחרנו הניב תוצאות יפות בהשוואה לדרכים אחרות עם היקף עבודה זהה ואף יותר רחב. הכלים והידע שרכשנו תוך כדי הפרויקט הקנו לנו את היכולת להתמודד עם בעיות לא צפויות ולהיות יצירתיים בפתירת בעיות בתחום, ונתנו לנו אינטואיציה לגבי שקלול גישות והשוואתן.

בנוסף נציין כי הפרויקט לימד אותנו בנוסף על עבודה מרחוק, שכן כל הפרויקט נעשה בתקופת סגרים והגבלות שלא אפשרו עבודה פנים אל פנים – לא עם המנחה ולרוב לא אחד עם השני. כלל מפגשי הפרויקט עם המנחה התקיימו בשיחת וידאו אך לדעתנו התגברנו גם על קושי זה בצורה טובה.

# בבליוגרפיה

AlexeyAB – darknet YOLOv4 <https://github.com/AlexeyAB/darknet>

# תודות

* תודה רבה לאיליה שהנחה אותנו וסיפק לנו את הכלים להרצת פרויקט זה
* תודה לפרופ' גיא גלבוע, יוסי בר ארז ויוחנן ארז על קידום הפרויקט והתמיכה המקצועית
* תודה לאינה טלמון, יוחנן ארז ודניאל יגודין על התמיכה הטכנית והמנהלתית לאורך הפרויקט
* תודה לפבל מהמעבדה של פרופ' מרק זילברשטיין שעזר למצוא את צוואר הבקבוק החישובי במערכת