

המחלקה להנדסת תוכנה

שם הפרויקט: אפליקציה לזיהוי ציפורים

ספר הפרויקט

שם הסטודנט: סתיו סוזן רבינוביץ

ליאור מטרני

שם המנחה: ד"ר שרון ילוב-הנדזל

מציע הרעיון/הצוות

מנחה וסטודנטים הקליני:

→ ☆ 17:57 (לפני 24 דקות)

Sharon Yalov-Handzel

אני, Lior

אישור המנחה על

בדיקת המסמך

ואישורו להגשה:

,ערב טוב סתיו

מאושרת הגשת ספר הנ

בהצלחה

...

תאריך ההגשה: 17/7/2022

תוכן עניינים

| | |
|----|--|
| 2 | תוכן עניינים |
| 5 | תודות |
| 6 | רשימות |
| 8 | תקציר הפרויקט |
| 10 | Executive Summary |
| 12 | מבוא |
| 13 | מטרות ויעדים |
| 13 | תיאור המטרות |
| 13 | יעדים פונקציונליים |
| 13 | כתיבת אלגוריתם ראייה ממוחשבת לזיהוי סוג הציפור לפי תמונה |
| 13 | הקמת יישומן סלולרי ייעודי לזיהוי התמונות ע"י מימוש האלגוריתם |
| 13 | הצגה ביישומן של ציפורים באזור לפי מיקום |
| 14 | הצגת מידע ביישומן אודות ציפור שזוהתה או אודות ציפור לפי בקשת המשתמש |
| 14 | מדדים |
| 15 | אופן השגת היעדים והמדדים |
| 16 | ריכוז שינויים |
| 17 | תכן החלופות שנבחרו למימוש |
| 17 | תיאור החלופה הפונקציונלית: פיתוח האלגוריתם – פיתוח באמצעות שילוב אלגוריתמים |
| 18 | תיאור החלופה הפונקציונלית: דרך פיתוח האלגוריתם – פיתוח האלגוריתם בשפת Python |
| 19 | תיאור החלופה הטכנולוגית: מימוש פלטפורמת SDK – פיתוח באמצעות React-Native |
| 20 | תיאור החלופה הטכנולוגית: שימוש בשירותי שרת / ענן – שימוש ב-Heroku |
| 20 | תיאור החלופה הטכנולוגית: שימוש במסד נתונים (DB) – שימוש ב-PostgreSQL |
| 21 | תרשימי ארכיטקטורה |
| 21 | תרשים ארכיטקטורה ברמת בלוקים |
| 23 | מבנה קשרי נתונים |
| 24 | תכן על |
| 25 | פירוט תכולת הפתרון |
| 25 | סקירת על של תוצר Alpha |
| 26 | דרישות מידע ופונקציונליות |
| 27 | פירוט מרכיבים ומודולים עיקריים |
| 27 | גישה לרשת |
| 27 | גישה למצלמה |
| 27 | גישה לאחסון בנייד |
| 27 | גישה למיקום |

| | |
|----|--|
| 27 | בסיסי נתונים בענן |
| 28 | פירוט והדגמת התכן, מימוש המערכת |
| 28 | אלגוריתם זיהוי התמונה – Bird Recognition Algorithm |
| 29 | הצגת תוצר Alpha, כפי שמומשה על בסיס התכנון במסמך דוח הביניים |
| 29 | תסריטים מבוססי Use-Cases עיקריים |
| 30 | חיפוש ציפור |
| 32 | זיהוי ציפורים |
| 34 | אימות צפר מקצועי |
| 35 | תיאור זרימת התהליכים העיקריים |
| 35 | הרשמה והתחברות |
| 35 | עדכון פרופיל |
| 35 | זיהוי ציפור |
| 35 | חיפוש ציפור |
| 35 | מבט על מיקומי ציפורים |
| 36 | אישור תמונות לאימון האלגוריתם |
| 37 | מסכי התהליכים העיקריים (ה-GUI) |
| 37 | מסך פתיחה |
| 38 | מסך רישום |
| 39 | מסך התחברות |
| 40 | תפריט ראשי |
| 41 | חיפוש ציפור |
| 42 | זיהוי ציפורים על מפה |
| 42 | צפייה ועדכון פרטי משתמש |
| 43 | זיהוי ציפור |
| 44 | מסך אימות זיהויים |
| 45 | ממשקים בין מודולים ותהליכים |
| 45 | התממשקות בין צד השרת לענן Heroku |
| 45 | התממשקות בין צד השרת לשרת אחסון התמונות ב-S3 |
| 46 | הדגמת קישור מודולים וארכיטקטורה |
| 47 | אלגוריתמים שמומשו במסגרת המערכת – אלגוריתם לזיהוי ציפורים |
| 47 | הקדמה |
| 47 | בחירת המודלים |
| 49 | אופן בניית האלגוריתם |
| 53 | ממצאי ביצוע בדיקות מערכתיות |
| 53 | בדיקות שבוצעו בעבור פונקציות ותרשימים |

| | |
|----|--|
| 54 | בדיקות שבוצעו בעבור היישומון |
| 56 | סיכום ומסקנות |
| 56 | שימוש במאגר תמונות רחב הינו קריטי לבניית האלגוריתם |
| 56 | תכנון ראוי ועמידה בלוחות הזמנים הינם מרכיב קריטי להצלחת הפרויקט |
| 57 | בעת חיבור בין מרכיבים (צד שרת ללקוח, אלגוריתם לצד שרת וכו') עלולות לעלות בעיות שלא היו קיימות לפניכן |
| 57 | תקשורת שותפת בין העובדים על הפרויקט חשובה להצלחתו |
| 58 | הצעה לעבודת המשך |
| 58 | שיפור האלגוריתם |
| 58 | פיתוח היישומון עבור חיות נוספות |
| 58 | הוספת אפשרות של זיהוי ציפור לפי קול |
| 59 | רשימת מקורות |
| 61 | נספחים |
| 61 | פוסטר הפרויקט – מצורף כמסמך נפרד |
| 61 | נספח א' – סקירה ספרותית ראשונית |
| 62 | מאמר 1: Bird Species Identification from an Image [1] |
| | מאמר 2: Look Closer to See Better: Recurrent Attention Convolutional Neural |
| 64 | [2] Network for Fine-grained Image Recognition |
| | מאמר 3: Analyzing the Performance of Apps Developed by using Cross- |
| 65 | [3] Platform and Native Technologies |
| | מאמר 4: CNN BASED CLASSIFIER FOR IDENTIFICATION OF CANINE |
| 66 | [4] BREEDS |
| 68 | נספח ב' – סקר שוק והשוואת מתחרים |
| 68 | יישומון [5] MerlinBirdID |
| 68 | אתר [6] BirdNet |
| 69 | אתר הצפרות הישראלי [7] |
| 69 | לקסיקון מפה: הציפורים - המדריך השלם לציפורי אירופה וישראל [8] |
| 70 | השוואה |
| 71 | נספח ג' – חלופות |
| 71 | חלופות טכנולוגיות למימוש |
| 71 | אלגוריתם לזיהוי ציפור |
| 73 | שפת פיתוח האלגוריתם |
| 75 | מימוש פלטפורמת SDK |
| 78 | שימוש בשירותי שרת / ענן |
| 79 | שימוש ב־DB |
| 80 | נספח ד' – תוכניות עבודה |

תודות

בשלב זה של הפרויקט חשוב לנו להודות תחילה לד"ר שרון ילוב-הנדזל, מנחת הפרויקט שלנו, על הליווי, הסיוע וההירתמות.

תודה נוספת שחשובה לנו במעמד זה, היא לפרויקט eBird, אשר אישרו לנו להשתמש במאגר תמונות הציפורים הקיים אצלם, לצרכי למידה ופיתוח היישומון שלנו במסגרת פרויקט הגמר.

1. רשימות

| סעיף | תיאור האירור |
|-------------|---|
| טבלה 6.1 | מדדים ואופן השגת יעדיהם |
| טבלה 7.1 | ריכוז שינויים |
| אירור 8.1 | תרשים בלוקים של המערכת |
| אירור 8.2 | מבנה קשרי נתונים |
| אירור 8.3 | תכן על של המערכת |
| אירור 8.4 | לוגו היישומון |
| אירור 10.1 | תרשים Use-case כללי |
| אירור 10.2 | מקרה של חיפוש ציפורים |
| אירור 10.3 | מקרה של זיהוי ציפורים |
| אירור 10.4 | מקרה של אימות צפר מקצועי |
| אירור 10.5 | מסך פתיחה |
| אירור 10.6 | מסך הרשמה |
| אירור 10.7 | מסך התחברות |
| אירור 10.8 | מסך התפריט הראשי (מנהל) |
| אירור 10.9 | מסך התפריט הראשי (משתמש רשום) |
| אירור 10.10 | מסך חיפוש ציפור |
| אירור 10.11 | מסך זיהוי ציפורים על המפה |
| אירור 10.12 | מסך צפייה בפרטי משתמש |
| אירור 10.13 | מסך עדכון פרטי משתמש |
| אירור 10.14 | מסך זיהוי ציפור |
| אירור 10.15 | מסך אימות זיהויים |
| אירור 10.16 | ארכיטקטורה של EfficientNet B0 |
| אירור 10.17 | ארכיטקטורה של DenseNet |
| אירור 10.18 | אישור לשימוש במאגר ע"י Jenna, נציגת eBird |
| אירור 10.19 | מתוך הקוד, יצירת Datan וה-generator לאימון, מבחן ואימות |
| אירור 10.20 | תוצאות אימון EfficientNet |
| אירור 10.21 | תוצאת האימון של DenseNet |
| אירור 10.22 | תוצאת האימון של VGG |

| | |
|-------------|---|
| איור 10.23 | מתוך הקוד, חיזוי תמונה |
| טבלה 11.1 | בדיקות פונקציות ותרחישים |
| טבלה 11.2 | בדיקות שבוצעו בעבור היישומן |
| טבלה 15.2.1 | מתוך מאמר [1], תוצאות האלגוריתמים הנבדקים במאמר |
| איור 15.2.2 | תמונה ממאמר [2], ניתוח תמונה ברשת נוירונים |
| איור 15.2.3 | תוצאות מאמר [3] |
| איור 15.2.4 | תמונה ממאמר [4], תמונות לדוגמה של גזעי כלבים |
| טבלה 15.2.5 | טבלה ממאמר [4], השוואת מודלים |
| טבלה 15.3.1 | השוואת מתחרים |
| טבלה 15.4.1 | השוואת חלופות אלגוריתם |
| טבלה 15.4.2 | השוואת חלופות דרכי פיתוח האלגוריתם |
| טבלה 15.4.3 | השוואת חלופות SDK |
| טבלה 15.5.1 | תאריכי הגשה לביצוע הפרויקט |
| טבלה 15.5.2 | לוח זמנים מפורט לפרויקט |

2. תקציר הפרויקט

מסמך זה מהווה את הדו"ח הסופי (Final Report) של פרויקט האפליקציה לזיהוי ציפורים. המסמך מציג את כל שלבי התכנון ההנדסי המפורט של הפרויקט, את תוצר ה Alpha ופיתוחו עפ"י התכנון, האילוצים והשינויים.

מטרת הפרויקט היא לתת מענה לחובבי צפרות ולצפרים ברחבי הארץ על ידי שימוש ביישומון שיודע לזהות ציפורים מתוך תמונה, ובנוסף, מאפשר לשמור את המיקום של אותה ציפור במאגר. מעבר לכך, משתמשי היישומון יוכלו לראות אילו ציפורים נמצאות באזור שלהם ולחפש מידע על ציפורים שונות. בדרך זו, זיהוי הציפורים יהיה זמין לכלל המתעניינים אשר יוכלו לקבל מידע נרחב על מיקומי ציפורים ולבצע מעקב נרחב יותר.

בדו"ח זה הסברנו על האלגוריתם של זיהוי הציפורים המהווה גרסה ראשונית של האלגוריתם למציאת ציפור לפי תמונה ומציאת התוצאה בצורה טובה. האלגוריתם מאפשר למשתמש לשלוח תמונה של ציפור ולפי מודל הקיים באלגוריתם שתוצאת האימון שלו המובילה, האלגוריתם יחזיר למשתמש מידע אודות אותה הציפור.

הדו"ח כולל תיאור החלופות בהן בחרנו להשתמש במהלך פיתוח היישומון, תוך מטרה להשיג יעילות, מהירות ואינטגרציה מרבית בין היישומון לממשקים, אשר יאפשרו ביצועים אופטימליים הן מבחינת ביצועי מערכת, והן מבחינות נוחות המשתמש.

עבור מימוש האלגוריתם נבחרה חלופה המשלבת ראייה ממוחשבת ורשתות נוירונים, בשל היתרונות של שילוב המודלים יחדיו.

בעבור פיתוח האלגוריתם נבחרה החלופה של שפת Python על פני ++C, R, Matlab כיוון שהשפה נוחה לשימוש ומאפשרת מימוש פשוט ושימוש בספריות ראייה ממוחשבת ולמידת מכונה.

בעבור מימוש צד הלקוח (SDK) נבחרה החלופה של React Native על פני Flutter, Android-Native ו-Native-IOS בשל האינטואיטיביות של השפה והיכולת להתרחב ולפתח את היישומון גם בעבור מכשירי IOS בקלות.

בעבור מימוש השרת נבחרה החלופה של Heroku עבור מסדי-הנתונים על פני שרת מקומי ו Amazon Web Services (AWS) בשל היכולת שלו להריץ אלגוריתמים, לקבל בקשות מהלקוח בכל זמן נתון מבלי שנצטרך להתעסק עם חיבור לרשת של השרת. כמוכן גם בזכות זמינותו והיכולת שלו להתממשק בנוחות ל-GitHub. את שרת Heroku נפעיל בשילוב עם שרת אחסון AWS S3 לתמונות כיוון שעבור אחסון התמונות, השרת של Heroku נתן תוצאות פחות מתאימות ושילוב שני השרתים מאפשר לפרויקט להתקדם בצורה הטובה ביותר.

בעבור מימוש מסד הנתונים (DataBase) אנו נבחר להשתמש ב PostgreSQL על פני MongoDB מכיוון שההתממשקות שלו מול שרתי Heroku יעילה הרבה יותר והכתיבה אליו נוחה.

מבחינת ניהול הסיכונים, הסיכונים ברובם נשארו זהים לסיכונים אשר חושבו בדו"ח ה-SOW.

הסיכון הקריטי ביותר הינו זיהוי לא נכון של ציפור קיימת. בשביל למזער את הסיכויים לקבלת סיכון זה, יש לאמן את האלגוריתם עד לתוצאת אימון רצויה ובנוסף, הוספת אימונים נוספים לאחר קבלת משאבים נוספים. בנוסף, מבוצעת בדיקת מספר מודלים לאימון המערכת לפני הכרעת האימון הטוב ביותר.

סיכון נוסף הינו בעיית התממשקות עם מסד נתוני המערכת, מה שעלול לגרום לאיבוד מידע חשוב כמו משתמשים רשומים, מאגר מידע של ציפורים, מאגר הציפורים של האפליקציה.. מה שפוגע באמינות היישומון. ניתן למנוע זאת ע"י בחירת סוג Database האופטימלי עבורנו, ווידוא אפשרות התממשקות

עם מסד נתונים זה עם השפה בה אנו משתמשים וסביבת הפיתוח שבחרנו. בנוסף, נגבה את המידע הקיים.

סיכון נוסף הוא בעיות התממשקות עם מערכת מפות חיצונית כיוון שהיא צריכה להציג נתונים על המפה. ניתן למנוע זאת בשלב התכנון בה הפונקציה שומרת את מיקומי הציפורים על המפה ותפקידנו לוודא שהפלטפורמה בה אנו מחליטים להשתמש מתאימה לדרישותינו.

בדו"ח זה הוספנו המלצות לעבודות המשך אפשריות לפרויקט זה ואת המסקנות שהועלו מהפרויקט.

Executive Summary .3

This report is the final report ("Project Book") of the FindBird Application Project, which is a mobile application for identifying birds from an image. The report presents all the engineering stages of the development and planning process with the aim of create the Alpha product.

The primary purpose of FindBird is to enable both amateur and expert birdwatchers to identify and learn about birds using a smartphone application. Furthermore, new bird locations (from recognitions) will be kept, allowing birdwatchers to discover which birds are in their region. In this approach, bird recognition will be available to all interested parties, allowing them to learn more about bird locations and track them.

In this report, we described the bird recognition algorithm, which is the alpha algorithm for discovering and identifying birds in an image and getting the best result. The algorithm allows the user to send an image of a bird, and by combining computer vision with neural net models, the algorithm will return to the application the bird's name, and the application will present information about this bird.

Additionally, this report refers to the alternatives that were used during our development to achieve efficiency, quickness, and the finest integration between both the application and the components, allowing optimal performance in both the system and the user experience.

The combination of computer vision and neural nets was picked as the algorithm development alternative since the benefits of the combination combined.

Python was chosen as an algorithm development language option over Matlab, C++, and R due to its ease of use and ability to quickly apply computer-vision and machine learning libraries.

Because of the language's intuitiveness and the ability to simply construct the application for both Android and IOS devices, React-Native was picked as the algorithm's SDK development option over Android-Native, IOS-Native, and Flutter.

Heroku's cloud server was chosen over local servers and Amazon Web Services (AWS) for the project's server alternative because of its ability to run algorithms, get and send requests from users at any time without us having to worry about network connections for the server all of the time, also because of its availability and integration with GitHub. We will utilize Heroku's server in conjunction with AWS S3 storage for the images, since Heroku's image storage produced less satisfactory results than their combination.

PostgreSQL was chosen as the project's database implementation choice over MongoDB owing to its ease of use and interaction with Heroku's server.

When it came to risk management, the majority of the risks remained the same as they were in the SOW report.

The most serious risk is incorrect identification of an existing bird. To lessen the likelihood of this danger, we must train the algorithm until we get the desired output. In addition, after receiving more bird images, further training will be added. Furthermore, there are model comparisons between several models, and the algorithm selects the model with the best training outcome.

Another concern is database integration issues, which can result in the leaking of essential information (such as registered users, bird information, bird locations, and so on), lowering the application's trustworthiness.

Another risk is database integration issues, which can result in the leaking of essential information (such as registered users, bird information, bird locations, and so on), reducing the application's trustworthiness. We may avoid that by deciding the proper database and checking the option to integrate with the language we write in and our development environment. In addition, we will make a backup of the existing data.

Another risk is issues with external map systems, which must display map features. We can prevent this at the planning stage, where the function preserves the birds' location on the map, and our responsibility is to ensure that the platform we choose fulfills our needs.

In this report, we included recommendations for ongoing projects as well as our project insights and realizations.

ישראל ממוקמת במעבר בין שלוש יבשות: אסיה, אירופה ואפריקה. לכן, במהלך השנה (ובעיקר בעונות הנדידה) ניתן לצפות במגוון ציפורים בעת נדידתן, עוברות מעל שמי הארץ.

כיום, בשביל הצפר החובב, מעקב מקצועי אחרי ציפורים מבוסס בעיקר על אתרי צפרות ברחבי הארץ וזיהוי של הציפורים נעשה על ידי צפרים מנוסים. לרוב הצפרים החובבים או אנשים המתעניינים, אין גישה מתמדת לאתרי צפרות ואין אפשרות להתייעץ עם צפר מנוסה בכל רגע נתון. כך נוצר מצב שהצפר החובב לא תמיד ידע לזהות את הציפור הנמצאת לידו – בטרם היא תעוף ותעלם.

בנוסף, עיקר הציפורים אליהן מתייחסים באתרי הצפרות הקיימים אלו ציפורים באותו האזור, וכן, צפרים מקצועיים אינם מאיישים את אתרי הצפרות בכל רגע נתון, מה שמונע מהם מעקב על ציפורים בכל רגע נתון, שאינו תלוי במיקומם הפיזי. מטרת הפרויקט היא לתת מענה לחובבי צפרות וגם לצפרים ברחבי הארץ על ידי שימוש ביישומון שידע לזהות ציפורים מתוך תמונה, וגם מאפשר לשמור את המיקום של אותה ציפור במאגר.

מעבר לכך, משתמשי היישומון יוכלו לראות אילו ציפורים נמצאות באזור שלהם ולחפש מידע על ציפורים שונות. בדרך זו, זיהוי הציפורים יהיה זמין לכלל המתעניינים וכן צפרים ואתרי צפרות יוכלו לקבל מידע נרחב על מיקומי ציפורים ולבצע מעקב נרחב יותר.

ביישומון נעשה שימוש באלגוריתם מבוסס ראייה ממוחשבת ורשתות נוירונים שידע לזהות את הציפור מהתמונה, וגם ישמור את נתוני הציפור לצורך אימונים נוספים שישפרו את איכות הזיהוי.

הפרויקט מחולק לשלושה חלקים - פיתוח אלגוריתם לזיהוי הציפורים, צד לקוח וצד שרת.

○ פיתוח האלגוריתם - עיקר המערכת, באמצעות רשתות נוירונים קיימות וספריות ראייה ממוחשבת, נפתח אלגוריתם שבסופו יקבל תמונה של ציפור וידע להחזיר למשתמש מידע אודות אותה הציפור.

○ צד לקוח וחווית המשתמש - בחלק זה נפתח את היישומון עצמו של המערכת, בחלק זה נתרכז בחלונות אותם נגיש למשתמש, נשים דגש על עיצוב ממשק וחווית שימוש של המשתמש.

○ צד שרת ובסיס הנתונים - בחלק זה הקשר בין צד הלקוח לאלגוריתם, נחבר בין היישומון למאגרי הנתונים ונפתח פונקציות נוספות אותן נרצה להציג ללקוח, בהן נשלב שימוש במאגרי הנתונים הקיימים לנו.

ביצוע העבודות בכל שלושת החלקים יבוצעו על ידי שני הסטודנטים בסנכרון מלא, אך לחלק מהתחומים יהיה סטודנט האחראי למעקב וניהול המשימות באותו התחום. חלוקת תחומי האחריות בפרויקט הותאמה לחוזקות, לידע הקיים הרלוונטי ולתחומי העניין של כל אחד מחברי הפרויקט.

סתיו סוזן רבינוביץ - התמחות בינה מלאכותית, אחראית על צד לקוח.

ליאור מטרני - התמחות בינה מלאכותית, אחראי על צד שרת.

פיתוח האלגוריתם עבור המערכת יהיה באחריות שניהם ובשיתוף פעולה מלא.

5. מטרות ויעדים

5.1. תיאור המטרות

מטרת הפרויקט היא לתת מענה לחובבי צפרות וגם לצפרים ברחבי הארץ על ידי שימוש ביישומון, המאפשר לזהות ציפור על ידי תמונה, לקבל מידע ומיקום של אותה ציפור במאגר. היישומון יאפשר לשתף נתונים על ציפור בין צפרים חובבים, מבקרי אתרי צפרות או מתעניינים.

5.2. יעדים פונקציונליים

1. כתיבת אלגוריתם ראייה ממוחשבת לזיהוי סוג הציפור לפי תמונה

איך השגנו?

תחילה, חיפשנו מאגר תמונות מספיק גדול בכדי שאלגוריתם יוכל ללמוד ממנו בצורה יעילה. לאחר מכן, בנינו ערכת נתונים (באנגלית: Dataset) ועל בסיס אותה ערכת נתונים בנינו את האלגוריתם. אימנו מספר מודלים: DenseNet201, VGG16, MobileNetV3 ו-EfficientNetB0. חילקנו את התמונות בערכת הנתונים ל-3 חלקים: אימון, מבחן ואימות (וואלידציה) וביצענו אימון על כל אחד מהמודלים. לאחר מכן, בחרנו במודל עם התוצאה הטובה ביותר ואימנו את המודל לפיו.

איך מדדנו?

לפי תוצאות המבחן במודלים, אנו יכולים לבחור במודל המתאים ביותר. כמובן שגם כאשר האלגוריתם עצמו יתאמן בהמשך, האימונים יתבצעו על שני המודלים והמודל שיבדוק את הציפור ביישומון יהיה המודל המדויק ביותר.

2. הקמת יישומון סלולרי ייעודי לזיהוי התמונות ע"י מימוש האלגוריתם

איך השגנו?

תחילה, למדנו להשתמש בטכנולוגיה של React-Native לבניית צד הלקוח ואת הטכנולוגיה של Flask בכדי ללמוד איך לחבר את האלגוריתם לצד השרת. באותו הזמן תוכננו המסכים הרלוונטיים שעתידיים להיות ביישומון.

בתום התכנון, נכתבו המסכים וצד השרת ולבסוף, חיברנו בין צד השרת לצד הלקוח, חיברנו בין האלגוריתם לצד השרת והקמנו בקשה מצד הלקוח לצד השרת שבעבור תמונה שהמשתמש מעלה – התמונה תעבור לאלגוריתם ומהאלגוריתם התשובה תחזור למשתמש.

איך מדדנו?

היישומון מתופעל באופן עצמאי ומשתמשי הקצה יכולים לבצע את הפעולה של צילום/ העלאת תמונה ולקבל בחזרה תשובה מהאלגוריתם איזו ציפור זוהתה בתמונה. לפיכך ניתן לראות כי המטרה שלנו הושגה.

3. הצגה ביישומון של ציפורים באזור לפי מיקום

איך השגנו?

בתוך היישומון הגדרנו בתוך המפה אפשרות להציג נקודות ציון של ציפורים (ושל שם המשתמש שצילם אותם).

איך מדדנו?

ציפורים עם מיקומים שהוגדרו במערכת הופיעו על גבי המפה. לפיכך ניתן לראות כי מטרה זו הושגה.

4. הצגת מידע ביישומון אודות ציפור שזוהתה או אודות ציפור לפי בקשת המשתמש

איך השגנו?

לכלל הציפורים הקיימות ביישומון קיים מידע בבסיס הנתונים של הפרויקט. כל משתמש יוכל לפנות למידע של אותה הציפור על-ידי בקשת משתמש פשוטה באפליקציה, או לחילופין – המידע יתקבל יחד עם התוצאה של זיהוי הציפור מתמונה.

איך מדדנו?

ניתן לראות כי ניתן לקבל מידע על כלל הציפורים המופיעות במערכת, לכן מטרה זו הושגה.

5.3. מדדים

1. לפחות 60% מהציפורים יזוהו לפי סוג הציפור
2. 75% מהציפורים יזוהו לפי משפחת הציפורים (יוניים, עגוריים, קוקייתיים וכו'..)
3. זיהוי ואימון האלגוריתם על לפחות 6 ציפורים מתוך 3 משפחות ציפורים שונות
4. ציון של 3 ומעלה (מתוך 5) בסקר שביעות רצון של המשתמשים ביישומון.

5.4. אופן השגת היעדים והמדדים

| מספר | מדד | יעד | מידת עמידה | סטייה | הסבר והארות |
|------|---|--------------------------------------|--------------|-------|---|
| 1 | כתיבת אלגוריתם ראייה ממוחשבת לזיהוי סוג הציפור לפי תמונה. | זיהוי של 60% מהציפורים | 0.815 | 15% | לפי האלגוריתם, המודל הטוב ביותר נכון לכתיבת דוח זה, מזהה את הציפורים ב-81.5% אך ככל שמספר התמונות והזיהויים ישתנה, מידת הדיוק יכולה להשתנות. אך כיוון שהאלגוריתם מתאמן תוך כדי – זו אינה בעיה. משקולות האלגוריתם נשמרות, כך שכל אימון אינו מתחיל מנקודת ההתחלה, אלא מנקודת האימון האחרונה. |
| 2 | | זיהוי של 75% ממשפחות הציפורים | | 10% | האלגוריתם זיהה וסיווג את הציפורים בצורה מספיק יעילה כך שלא היה צורך בזיהוי משפחת ציפורים. |
| 3 | | זיהוי 6 ציפורים ממשפחת ציפורים שונות | נעמד | אין | האלגוריתם מזהה 6 ציפורים מ-3 משפחות ציפורים שונות |
| 4 | הקמת יישומון סלולרי ייעודי לזיהוי התמונות ע"י מימוש האלגוריתם | דירוג משתמשים של +3 מתוך 5 | עמידה משוערת | אין | מתוך המשתמשים להם אפשרנו להתנסות ביישומון, הדירוג היה 4.5 אך מספר המשתמשים היה נמוך, כך שהתוצאה אינה מדויקת. ככל שיהיו יותר משתמשים כך התוצאה תהיה מדויקת יותר |
| 5 | הצגה ביישומון של ציפורים באזור לפי מיקום | | | | |
| 6 | הצגת מידע ביישומון אודות הציפור שזוהתה או לפי בקשת המשתמש. | | | | |

טבלה 6.1 מדדים ואופן השגת יעדיהם

6. ריכוז שינויים

| שינוי | סיבה | השפעה על התוכנית | משמעויות נוספות |
|--|--|--|--|
| שינויים שהתרחשו לפני דו"ח הביניים | | | |
| שימוש בשירות ענן Heroku | כיוון שלפני שלב זה טרם נקבעה צורת המימוש לשרת בעבור היישומון, החלטנו להשתמש בשרת הענן של Heroku. מאפשרת לנו להריץ את הקוד בענן בצורה חנימית. ובנוסף Heroku נותן לנו גם להריץ כ-SASS כך שדבר זה חסך לנו הקמה של שרת DB רלציוני וקונפיגורציה שלו. | אין ליישומון תלות בשרת פיזי. ניתן להתחבר לשרת מכל עמדת קצה (מכשיר חכם) המחובר לרשת האינטרנט. | Heroku מאפשר ביצוע Deploy לתוכנית שאנו כותבים בצורה נוחה יעילה וקלה יותר. |
| שימוש בספריית Flask של Python לצד השרת | בחרנו להשתמש ב-Flask כיוון שהכתיבה של האלגוריתם היא בשפת Python, מה שמהווה יתרון גם להתחברות וההתממשקות של האלגוריתם לזיהוי, וגם בשל הניסיון של שנינו בשפה לאורך הלימודים. סיבה נוספת היא העובדה כי Flask ספרייה נפוצה בעל קהילה גדולה שיכולה לעזור ולתמוך. | אין השפעה כלשהי. | ההתממשקות בין האלגוריתם לשרת יהיה נוח יותר בזכות השימוש באותה השפה. |
| הגדלת מאגר הנתונים | במאגר הנתונים שאנו משתמשים ישנם רק 30 תמונות לכל ציפור, כמות זו אינה מספיקה כדי לפתח אלגוריתם עם אחוז זיהוי גבוה. לכן אנו מגדילים את מאגר הנתונים בעזרת שימוש של מאגרי נתונים חיצוניים. לכן, פנינו לאתר הצפרות הישראלי והם נתנו לנו גישה לתמונות נוספות של ציפורים בארץ. | בניית מאגר נתונים גדול יותר לטובת אימון האלגוריתם. | ככל שנגדיל את מאגר הנתונים שלנו, כך האלגוריתם יכול יותר להתאמן על מגוון רחב יותר של תמונות ציפורים ולהשתפר בתוצאות. |
| שינויים שהתרחשו לפני הדו"ח המסכם | | | |
| הוספת AWS S3 כשרת למאגר נתונים | בחרנו להשתמש ב-AWS S3 לתמונות כיוון שהשרת של Heroku לא התאים לאחסון תמונות. | שינוי מיקום אחסון התמונות שהמשתמשים מעלים ליישומון. | בעת בקשת תמונה של ציפור יש לגשת גם ל-S3. בשרת זה אנו נשמור תמונות להצגה למשתמש על כל ציפור וגם כן את התמונות שהמשתמשים צילמו על מנת שנוכל להשתמש בתמונות אלו לאמן את האלגוריתם שלנו. |
| הורדת זיהוי הציפור לפי משפחה מהאלגוריתם | הורדת משפחת הציפורים כיוון שיש תוצאה טובה של זיהוי ציפור כציפור בעצמה, ואימון כפול שכזה רק יכביד על האלגוריתם ולא ישפר את תפקודו. | האלגוריתם לא צריך לזהות פעמיים כל ציפור, ולכן רץ מהר יותר | אין צורך בשליחת משפחת הציפורים בין טבלאות ב-DB. |

טבלה 6.1 ריכוז שינויים

7. תכן החלופות שנבחרו למימוש

החלופות הפונקציונליות והטכנולוגיות אשר נבחרו מפורטות להלן. בכדי לקרוא את השוואת כלל החלופות ואת החלופות שאינן נבחרו, יש לעיין בנספח ג'.

7.1 תיאור החלופה הפונקציונלית: פיתוח האלגוריתם – פיתוח באמצעות שילוב אלגוריתמים

אלגוריתם הזיהוי מהווה חלק משמעותי ובלתי נפרד מהפיתוח של המערכת. ללא זיהוי נכון של הציפורים - המערכת תפספס את המטרה המרכזית שלה. לכן חשוב לבחור בחלופה המתאימה ביותר לזיהוי המבוקש.

בחלופה זו, האלגוריתם שפותח ישלב אלמנטים מאלגוריתמים מבוססי תכונות התמונה ואלגוריתמים מבוססי ראייה ממוחשבת ולמידה עמוקה. בדרך זו אנו מאמנים בדרכים שונות את המאגר ומשלבם את היתרונות שבשתי השיטות.

רכיבים רלוונטיים

חיבור למאגר מידע, ספריות OpenCV, התקנת Python על המחשב.

אופן מימוש

כתיבת אלגוריתם המתבסס גם על מודלים של עיבוד תמונה וגם על מודלים של למידה עמוקה.

רמת קושי למימוש

מימוש מורכב יותר ביחס לחלופות האחרות

יתרונות

שילוב היתרונות שבאלגוריתמים מבוססי תכונות תמונה ובאלגוריתמים מבוססי ראייה ממוחשבת ולמידה עמוקה: התאמת האלגוריתם במדויק לצרכים שלנו, לפי הקריטריונים עליהם אנחנו רוצים להתבסס, וגם במימוש והשוואה בין מספר מודלים התוצאה תהיה מדויקת יותר.

חסרונות

זמן אימון ארוך מכל שיטה בנפרד

7.2. תיאור החלופה הפונקציונלית: דרך פיתוח האלגוריתם – פיתוח האלגוריתם בשפת Python

השימוש חינוכי, בPython קיימות ספריות רבות למתכנתים שרוצים לפתח בסביבה זו אלגוריתמים של ראייה ממוחשבת, כדוגמת Keras, Pytorch. בנוסף, קיימות ספריות כמו OpenCV לצורך עריכת תמונה. ממשק נוח בצורת קוד בלבד. מפתחים רבים בוחרים לממש ולפתח קוד בשפה זו.

ממשק משתמש

קוד בלבד.

מחיר

חינמי.

תמיכה בספריית OpenCV

קיימת תמיכה.

סביבות פיתוח אפשריות

Jupyter Notebook, Google Collab.

נוחות כתיבת הקוד

נוחות מרבית.

תמיכת הסביבה בGPU

קיימת תמיכה.

רמת קושי למימוש

קל למימוש.

7.3. תיאור החלופה הטכנולוגית: מימוש פלטפורמת SDK – פיתוח באמצעות React-Native

פלטפורמת React Native נכתבת ב-Java Script ופותחה ע"י חברת Facebook. פלטפורמה זו אינטואיטיבית לבניית ממשק משתמש עבור מפתחים ומאפשרת מגוון אפשרויות פיתוח.

פלטפורמה זו מאפשרת לנו לפתח במקביל יישומון למכשירים חכמים בעלי מערכת הפעלה Android וגם למכשירים חכמים בעלי מערכת הפעלה IOS. דהיינו, פיתוח במקביל לשתי מערכות ההפעלה, מה שמקצר בהרבה את שלבי הפיתוח לעומת פיתוח יישומון נפרד לכל מערכת הפעלה.

סביבות פיתוח

Visual Studio Code, Android Studio (emulator).

תמיכה במכשירי אנדרואיד

קיימת תמיכה.

תמיכה במכשירי IOS

קיימת תמיכה.

שפות תכנות

Java Script.

רמת קושי למימוש

מימוש פשוט

יתרונות

- פיתוח עתידי למכשירים חכמים תומכי IOS, יתאפשר ללא פתיחת פרויקט חדש – אנו נבצע התאמות בקוד הקיים.
- הפלטפורמה מודולרית ואינטואיטיבית.
- קהילה המשתמשת בפלטפורמה זו רחבה וקיימת תמיכה רבה.
- תמיכה של Meta.
- אפשרויות עיצוב מגוונות.

חסרונות

- איתור השגיאות מסובך.
- סביבה יחסית חדשה, קיימים עדכונים ושינויים רבים בסביבה בפרק זמן קצר.

7.4. תיאור החלופה הטכנולוגית: שימוש בשירותי שרת / ענן – שימוש ב-Heroku

עבור צד השרת, נרצה שרת שיוכל להריץ את האלגוריתמים שלנו, לקבל בקשות מהלקוח בכל זמן נתון. ושלא נצטרך להתעסק עם חיבור לרשת של השרת, זמינות של הרשת אצלנו בכל רגע נתון. משום כך, נשתמש בשרת ענן שיוכל לעשות לנו זאת. אנו בחרנו להשתמש בשרת Heroku שיש לו התממשקות טובה עם Github שם אנו מנהלים את ניהול הגרסאות של הפרויקט.

יתרונות

- מאוד פשוט למימוש, קל לבצע עדכון בסביבת הפיתוח
- מאובטח בעזרת טכנולוגיות אבטחת המידע המעודכנות ביותר
- בעזרת Containers של Heroku (נקראים dynos), לא צריך להתעסק בהרמת השרת, הם עושים זאת.

חסרונות

- איטי יותר מחלק מסביבות הענן בשוק.
- לא ניתן לקנפג את כל הגדרות השרת, כלומר – אין יכולת לשלוט על השרת במלואו.

7.5. תיאור החלופה הטכנולוגית: שימוש במסד נתונים (DB) – שימוש ב-PostgreSQL

בסיס הנתונים איתנו אנחנו משתמשים כרגע הוא PostgreSQL שהוא בסיס נתונים רלציוני המפותח בתור פרויקט "קוד פתוח".

יתרונות

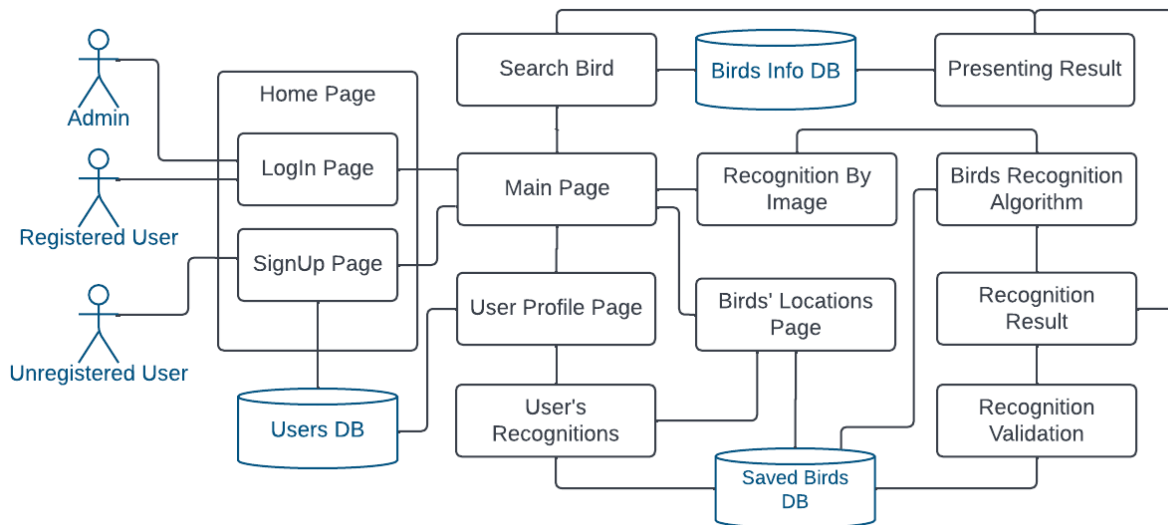
- מציג נתונים סטטיסטיים על שימוש בשרת.
- מאוד קל ללמוד להשתמש.
- מתרגם בקשות מהקוד ל SQL בקלות, מפשט את כתיבת שאילתות.
- גישה לתוספים, דרכים נוספות לעבוד עם המידע.

חסרונות

- רוב תוכניות הקוד הפתוח אינן תומכות PostgreSQL.
- איטי מחלק מבסיסי הנתונים הרלציוניים בשוק.

7.6. תרשימי ארכיטקטורה

7.6.1. תרשים ארכיטקטורה ברמת בלוקים



איור 8.17 תרשים בלוקים של המערכת

באיור 8.1 ניתן לראות את תרשימי הבלוקים המתאר את ארכיטקטורת המערכת.

נכון להגשת מסמך זה, לא קיימים שינויים במודולים הקיימים במסמך ה-SOW.

1. **מאגר מידע ציפורים (Birds Info DB) –** מאגר אשר יכיל את כל המידע על הציפורים שיופיע למשתמש כאשר יחפש אותו.

2. **מאגר משתמשים (Users DB) –** המאגר בו ישמרו הנתונים אודות כל המשתמשים הרשומים במערכת, כלומר המשתמשים הרגילים, מנהלים.

3. **מאגר ציפורים מזוהות (Saved Birds DB) –** מאגר הציפורים שאומתו במערכת. במאגר זה ישתמשו לאחסון התמונות שזוהו, מיקומי הציפורים וגם כתמונות לאימון האלגוריתם כל תקופה.

4. **מודול מסך בית (Home) –** מסך בית, דרכו ניתן לגשת למודול ההרשמה ולמודול ההתחברות למערכת.

5. **מודול הרשמה (Sign-up) –** מסך הרשמה למערכת, כל משתמש רגיל ולא רשום יכול להירשם למערכת משם. ההרשמה תאפשר למשתמש להזדהות בפעמים הבאות בהן יתחבר ולגשת למודולים המאפשרים גישה רק למשתמשים רשומים או למנהלים. בנוסף, לאחר ההרשמה הנתונים שהמשתמש יעלה, יישמרו לעתיד.

6. **מודול התחברות (Sign-in) –** מסך התחברות למערכת, משם יתחברו למערכת גם משתמשים רגילים וגם מנהלים. ההתחברות מאפשרת גישה למודולים המוגדרים למשתמשים רשומים ולמנהלים. בנוסף, הנתונים שהמשתמש יעלה, יישמרו לעתיד.

7. **מודול ראשי (Main Page) – המסך הראשי**, ממנו תינתן הגישה לכלל המודולים הקיימים ביישומן.

8. **מודול פרופיל משתמש (User Profile) – מסך שבו המשתמש יכול לראות את ההגדרות שלו**, לשנות, ולעדכן את המשתמש.

9. **מודול זיהוי משתמש (User's Recognitions) – מסך בו יוצגו כל זיהויי הציפורים שהמשתמש עושה ביישומן שלו**, המודול יפעל דרך מודול מיקומי ציפור, בעבור הזיהויים של משתמש ספציפי.

10. **מודול חיפוש ציפור (Search Bird) – מודול זה יבקש מהמשתמש איזה ציפור הוא רוצה**. במידה והמשתמש יחליט לחפש שם של ציפור, המערכת תעביר ממאגר הנתונים את המידע עבור הציפור אותה המשתמש מבקש למודול תוצאת החיפוש.

11. **מודול תוצאת חיפוש (Presenting Results) – במודול זה יוצג למשתמש מידע אודות הציפור שעלתה בחיפוש או עלתה כתוצאה של האלגוריתם**.

12. **מודול מיקומי ציפור (Bird's Locations) – מודול המציג את מיקומי הציפורים שנראו לאחרונה**. המערכת תציג מספר תוצאות אחרונות שהיו להצגת הציפור. ניתן לשנות את טווח הזמנים הרלוונטי.

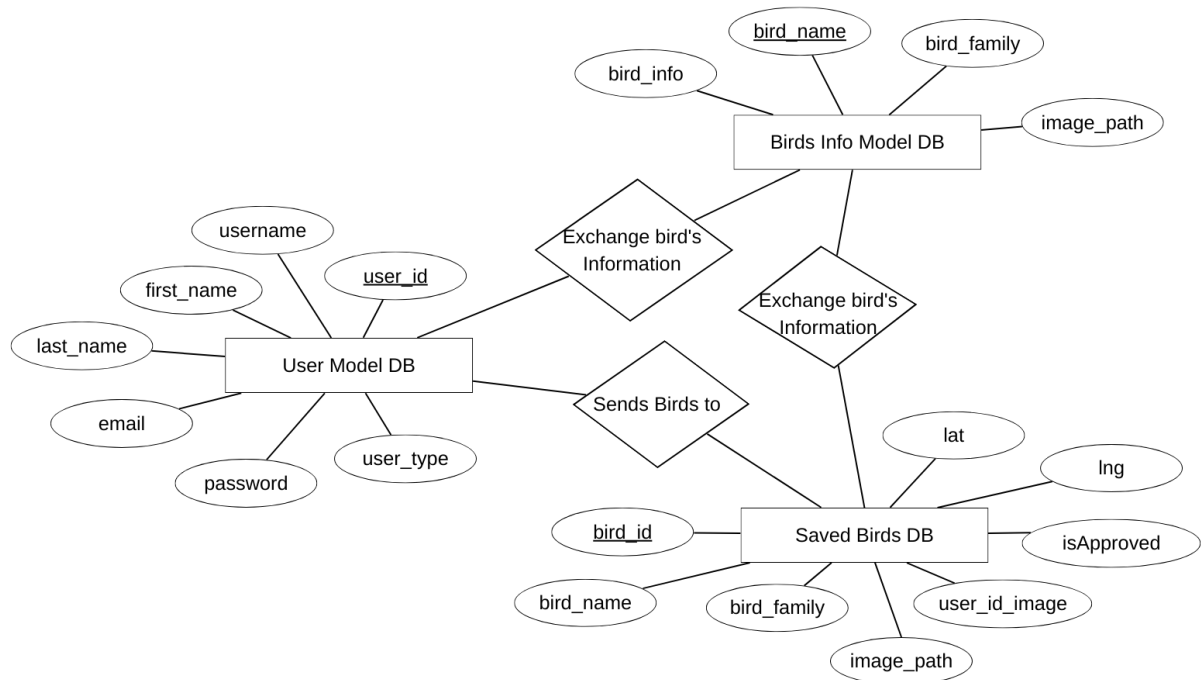
13. **מודול זיהוי לפי תמונה (Recognition By Image) – המודול יאפשר להעלות תמונה דרך האחסון במכשיר החכם או דרך המצלמה**, בה יוכל המשתמש לבחור לצלם את התמונה ואם התמונה לא יצאה ברורה, לצלם מחדש. מודול זה מאפשר לשלוח את התמונה לתהליך הזיהוי של התמונה לפי האלגוריתם.

14. **אלגוריתם לזיהוי ציפורים (Birds Recognition Algorithm) – האלגוריתם אותו נבנה לצורך זיהוי הציפורים**. אלגוריתם זה הוא חלק מתוצרי הפרויקט.

15. **מודל תוצאת אלגוריתם (Recognition Result) – במודול זה נקבל את תוצאת הזיהוי**, פרטי הזיהוי יועברו למאגר המיקומים והמודל יעביר את המשתמש למסך הצגת התוצאה.

16. **מודול אימות זיהויים (Recognition Validation) – כדי לאפשר לציפורים שזוהו ותמונותיהן להצטרף למאגר האימון**, מנהל המערכת יוכל באמצעות מודול זה לאמת תוצאות ולהעביר את נתוניהם למאגר הציפורים של האלגוריתם.

7.6.2. מבנה קשרי נתונים

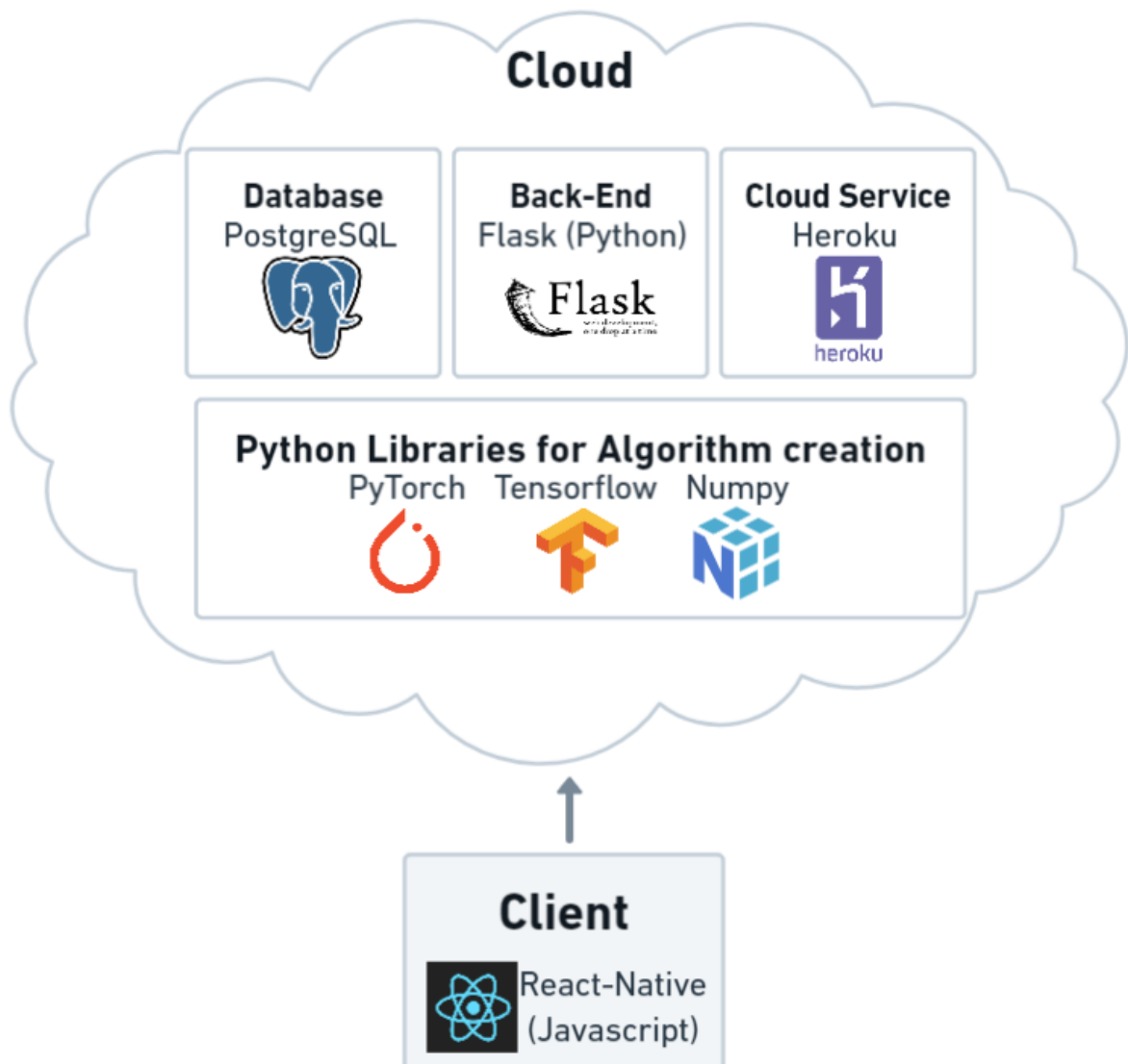


איור 8.27 מבנה קשרי נתונים

השחקן הראשי במערכת, המשתמש, מיוצג ע"י מספר מזהה, המזהה אותו באופן חד-חד ערכי (חח"ע). המשתמש יוכל להתחבר למערכת ע"י שימוש בסיסמה אותה בחר.

הציפורים השמורות אלו אותן התמונות שהמשתמשים העלו בכדי לזהות את הציפור בה. לכל ציפור שכזו יש מספר מזהה, המזהה אותה באופן חח"ע. במידה ונרצה להוסיף את ציפור זו למאגר התמונות לאימון, משתמש מסוג מנהל יאשר אותן.

מידע על ציפור נשמר על פי שם הציפור, כשציפור בתמונה מזוהה כציפור שמורה במערכת, המערכת מציגה למשתמש את המידע אודות אותה ציפור. כמובן, כאשר משתמש מבקש.



איור 8.37 תכן על

כפי שנאמר והוחלט בסעיפים הקודמים, עבור צד הלקוח (Client) נשתמש ב-React-Native. מבחינת צד הענן (שרת) אנו נשתמש בטכנולוגיית הענן של Heroku, במסד נתונים תומך PostgreSQL, את צד הלקוח אנו נתכנת ב-Flask, ועבור האלגוריתם אנו נשתמש ב-PyTorch, Numpy ו-TensorFlow.

7.7. פירוט תכולת הפתרון

7.7.1. סקירת על של תוצר Alpha



תוצר Alpha של הפרויקט הוא יישומון למכשירים חכמים בשם "Find Bird". מטרת היישומון היא לאפשר לחובבי ציפורים, צפרים חובבים או צפרים מקצועיים לקבל זיהוי בזמן אמת על הציפור אותה הם מצלמים ובנוסף, לעקוב אחרי מיקומי ציפורים באזור שלהם.

בכדי להשיג את מטרות אלו, ולאפשר ליישומון לפעול כראוי, פיתחנו אלגוריתם לזיהוי הציפור.

איור 8.4 לוגו היישומון

האלגוריתם מקבל מהיישומון תמונה (ע"י צילום של היישומון או תמונה שהועלתה דרכו) של ציפור, הוא מעבד את התמונה ובוחן אותה ע"י האימונים אותם ביצע ומחזיר למשתמש את שם הציפור ומידע אודותיה.

בנוסף, בכל פעם שמשתמש מעלה ציפור ליישומון, פרטי הציפור (הצלם, מיקום הצילום וכו') נשמרים יחד עם התוצאה (שם הציפור שזוהתה) וכך המשתמשים נהנים מצפייה במגוון הציפורים באזוריהם.

7.7.2. דרישות מידע ופונקציונליות.

1. תחת תוצר Alphan ימומשו מערכות הרשמה והזדהות, כך שהמידע על המשתמשים והציפורים שיזהו יישמר בענן.
2. מימוש מנגנון ההתחברות ע"י כתובת מייל וסיסמה.
3. מסך פרופיל בו המשתמש יוכל לראות את פרטיו ולעדכן אותם.
4. מסך חיפוש ציפור, בו המשתמש יוכל לקרוא אודות ציפור הקיימת במערכת ואותה מחפש.
5. מסך זיהוי ציפורים, בו המשתמש מצלם או מעלה תמונה של ציפור.
6. מנגנון לאיסוף תמונת המשתמש ופרטי התמונה, לצורך שליחה לאלגוריתם ושמירת התוצאה והתמונה בשרתי המשתמש.
7. מסך זיהוי ציפורים באזור, המראה מיקומי ציפורים באזור המשתמש.
8. מסך אישור מנהל, בו מנהל המערכת (משתמש הרשום Admins) יוכל לאשר זיהויים ובמידת הצורך גם לעדכן את פרטיהם, לפני שליחתם כתמונות לאימון האלגוריתם.

7.8. פירוט מרכיבים ומודולים עיקריים

7.8.1. גישה לרשת

לצורך הרשמה, הזדהות, שליפת נתונים וזיהוי ציפור חדשה, המשתמש יצטרך להתחבר לשרתים בענן. בכדי להתחבר ולהשתמש בנתונים המצויים בענן, על המשתמש לדאוג לחיבור ישיר לרשת האינטרנט.

נכון לכתיבת דוח זה, בעת שימוש במכשיר חכם אין צורך באישור יזום של המשתמש לשימוש ברשת האינטרנט (סלולרית או Wifi) והגישה מתאפשרת כברירת מחדל.

7.8.2. גישה למצלמה

בשביל לשלוח תמונה לאלגוריתם דרך היישומון, החלופה הראשונה היא דרך צילום ביישומון. בעבור צילום תמונה דרך היישומון, נדרשת גישה למצלמה. על המשתמש לאפשר ליישומון שימוש בגישה למצלמה דרך מערכת ההפעלה. או לחילופין, לאפשר מיקום לאחסון במכשיר (סעיף 8.8.3).

7.8.3. גישה לאחסון בנייד

בשביל לשלוח תמונה לאלגוריתם דרך היישומון, החלופה נוספת היא העלאת תמונה קיימת מהנייד, ובעבור כך, נדרשת גישה לאחסון במכשיר. על המשתמש לאפשר ליישומון גישה לאחסון במכשיר דרך מערכת ההפעלה. או לחילופין, לאפשר גישה למצלמה במכשיר (סעיף 8.8.2).

7.8.4. גישה למיקום

לצורך זיהוי מיקום המשתמש (מקום צילום של תמונת הציפור, או ציפורים שזוהו באזור המשתמש), נדרשת גישה ל-GPS המותקן בכל מכשיר חכם – כלומר, נדרשת גישה למיקום המכשיר החכם. על המשתמש לאפשר גישה למיקום המכשיר דרך מערכת ההפעלה.

7.8.5. בסיסי נתונים בענן

כמפורט בסעיף 8.4, מבוצע שימוש ברשת בכדי ליצור קשר עם האלגוריתם ועם בסיסי הנתונים שבענן.

8. פירוט והדגמת התכן, מימוש המערכת

תוצר הפרויקט הסופי הינו היישומון, **"Find Bird"**, שמטרתו העיקרית הנה לאפשר לחובבי ציפורים, צפרים חובבים או צפרים מקצועיים לקבל זיהוי בזמן אמת על הציפור אותה הם מצלמים. האלגוריתם העיקרי עליו מתבסס היישומון הוא האלגוריתם לזיהוי ציפור לפי תמונה.

במהלך השימוש ביישומון, המשתמש אינו מבחין בהרצת האלגוריתם עצמו, שכן הרצתו מתבצעת בענן מרוחק, ולא על המכשיר עצמו.

הסיבה המרכזית שעבורה היישומון לא מריץ את האלגוריתם בעצמו, היא שאימון האלגוריתם והרצתו יכולים לתפוס שטח אחסון והרצה (GPU) רבים ובכך להחליש את המכשיר בעת שימוש ביישומון. סיבה נוספת היא חוסר הסנכרון שיווצר בין משתמשים שונים באלגוריתם. ככל שקבוצת האימון גדולה יותר ומגוונת יותר, כך האלגוריתם יזהה בצורה מדויקת יותר.

8.1 אלגוריתם זיהוי התמונה – Bird Recognition Algorithm

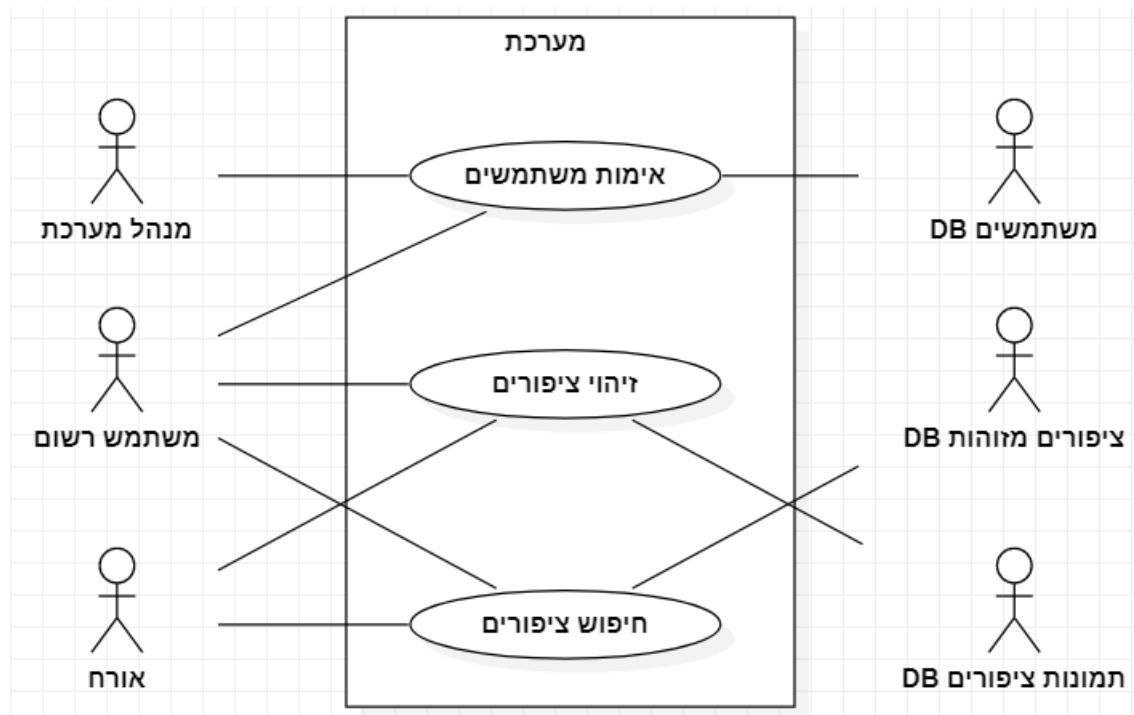
אלגוריתם הזיהוי מזהה את הציפור הנמצאת בתמונה. האלגוריתם מתבסס על עיבוד תמונה ורשתות נוירונים.

בשלב הראשון המשתמש מעלה תמונה (ע"י צילום / העלאת תמונה קיימת מהמכשיר) ליישומון. היישומון שולח את התמונה מהמשתמש לאלגוריתם. האלגוריתם מקבל את התמונה, מעבד אותה ולבסוף – האלגוריתם מחזיר ליישומון את שם הציפור שזוהתה. היישומון מבקש מהשרת נתונים אודות אותה הציפור, וכלל המידע מוצג למשתמש.

פירוט נוסף על האלגוריתם ניתן למצוא בסעיף 10.6.

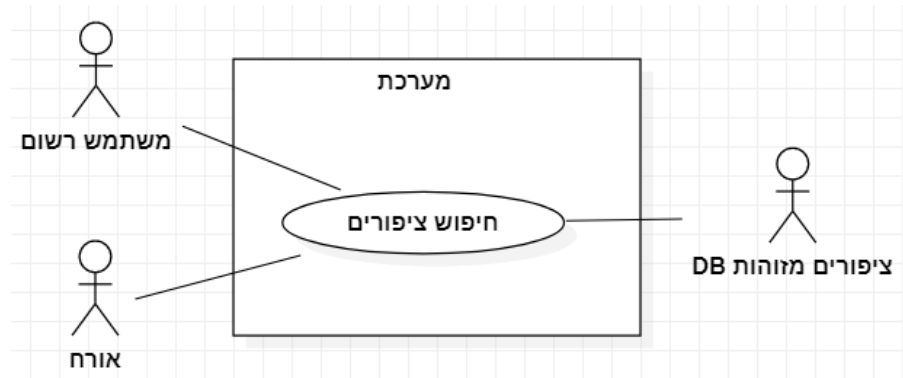
9. הצגת תוצר Alpha, כפי שמומשה על בסיס התכנון במסמך דוח הביניים

9.1. תסריטים מבוססי Use-Cases עיקריים



איור 10.1 תרשים Use-case כללי

9.1.1. חיפוש ציפור



איור 10.29 מקרה של חיפוש ציפורים

זרימה ראשית

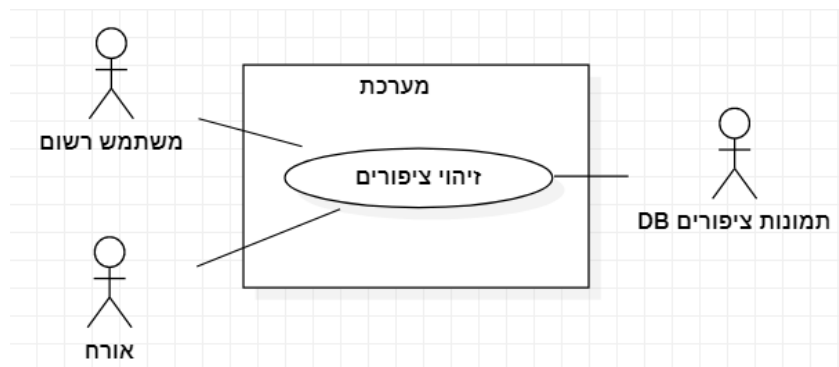
1. המשתמש נכנס למערכת (כאורח או כמשתמש)
2. המערכת שואלת את המשתמש שאלות על הציפור אותה מחפש לפי שם, לפי מיקום או לפי משפחת ציפורים.
3. המשתמש מזין את שם הציפור אותה מחפש, או עונה על לפחות אחת מהשאלות בבחירת המאפיינים.
4. המערכת מחפשת מתוך כלל הציפורים בDB את הציפורים העונות על אותן השאלות.
5. המערכת מציגה למשתמש את הציפורים העונות על אותן הקריטריונים שהציג.
6. המשתמש בוחר את הציפור המבוקשת.
7. המערכת מציגה מידע אודות אותה הציפור.

זרימה משנית א

תרשים זרימה משנית בעקבות סעיף 4 בזרימה הראשית

1. המערכת בודקת אם הוזן שם - במידה ולא הוזן, עוברים לשלב 6 בזרימה משנית זו.
2. המערכת מחפשת במאגר הציפורים ציפור בעלת אותו שם.
3. אם המערכת מוצאת ציפור בעלת אותו שם - המשך בזרימה הראשית בשלב 7.
4. אם לא נמצאה ציפור בעלת אותו השם - המערכת שולחת הודעה למשתמש שלא נמצאה ציפור בשם זה.
5. המשך לשלב 3 בזרימה הראשית.
6. המערכת בודקת אם נבחרו משפחות ציפורים - במידה ולא נבחרו, עוברים לשלב 9 בזרימה משנית זו.
7. המערכת מחפשת את כל הציפורים במערכת מאותן משפחות ציפורים.
8. אם המערכת מוצאת ציפורים מאותה המשפחה - המשך בזרימה הראשית בשלב 5.
9. המערכת בודקת מהו מיקום המשתמש ומה הרדיוס אותו בחר - במידה ולא הוזן, עוברים לשלב 15 בתרשים זה.
10. המערכת מחפשת את שמות הציפורים שנמצאו ברדיוס של אותו המיקום במערכת.
11. המערכת מחפשת במאגר המידע את הציפורים עם אותם השמות.
12. אם המערכת מוצאת ציפורים/אלו - המשך בזרימה הראשית בשלב 5.
13. במידה ולא נמצאו ציפורים ברדיוס זה - המערכת שולחת הודעה למשתמש שלא נמצאו ציפורים ברדיוס הנבחר במיקום הנוכחי.
14. המשך לשלב 3 בזרימה הראשית.
15. המערכת מתריעה שלא נבחרה דרך חיפוש.
16. המשך לשלב 3 בזרימה הראשית.

9.1.2. זיהוי ציפורים



איור 10.39 מקרה של זיהוי ציפורים

זרימה ראשית

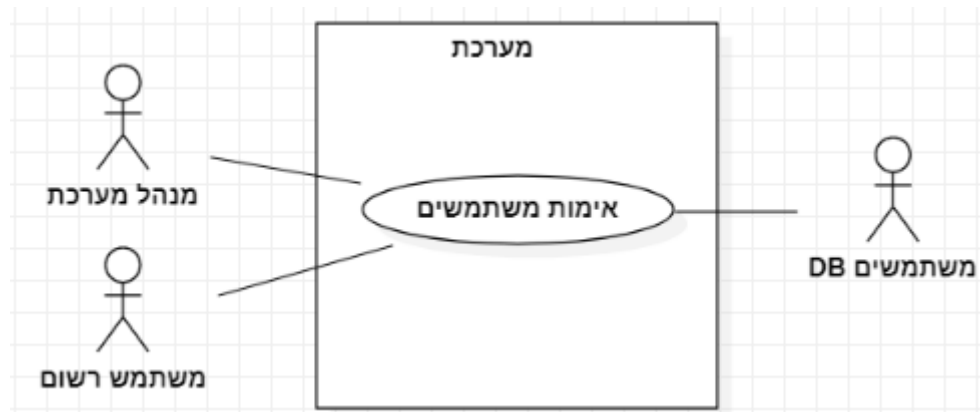
1. המשתמש נכנס למערכת ומבקש לזהות ציפור.
2. המערכת מבקשת מהמשתמש מידע על המיקום שלו ותמונה של הציפור.
3. המשתמש מאפשר קבלת מיקום ומעלה למערכת תמונה של הציפור.
4. התמונה נשלחת לזיהוי אצל האלגוריתם.
5. האלגוריתם מנקה את התמונה מרעשים.
6. האלגוריתם שולח את התמונה למודל המאומן לקבלת תוצאה.
7. האלגוריתם מחזיר תשובה.
8. המערכת שולחת את פרטי הזיהוי (תמונה, מיקום ונתונים) לאימות.
9. המערכת מחזירה למשתמש מידע אודות הציפור שזוהתה.

זרימה משנית א

זרימה משנית בעקבות סעיף 7 בזרימה הראשית:

1. האלגוריתם בודק אם התוצאה המיטבית של שם הציפור היא ברמת זיהוי של מעל 40% לאותה הציפור.
2. במידה וקיימת ציפור כזו, עבור לשלב 8 בתרשים הזרימה הראשי.
3. האלגוריתם בודק אם התוצאה המיטבית של משפחת הציפורים היא ברמת זיהוי של מעל 50% לאותה הציפור.
4. במידה וקיימת משפחת ציפורים כזו, החזר מידע אודות משפחת הציפורים שזוהתה ומידע על הציפורים באותה המשפחה.
5. במידה והמשתמש רשום (כצפר מקצועי) המערכת תאפשר למשתמש לבחור איזו ציפור מאותה משפחה מתאימה לזיהוי זה
6. המשך לשלב 9 בתרשים הראשי

9.1.3. אימות צפר מקצועי



איור 10.49 מקרה של אימות צפר מקצועי

זרימה ראשית

1. משתמש מציין בזמן הרישום שהוא צפר מקצועי.
2. המערכת מבקשת ממנו תמונה/ מסמך של תעודת הסמכה.
3. המשתמש מעלה למערכת.
4. המשתמש נרשם כמשתמש רגיל ורק לאחר אישור מנהל המערכת יסומן כצפר מקצועי.
5. מנהל המערכת נכנס לאישור משתמשים במערכת.
6. המערכת מציגה למנהל המערכת את פרטי הרישום ואת תעודת ההסמכה של הצפר.
7. מנהל המערכת מחליט אם לאשר או לדחות את הבקשה.
8. במידה ומנהל המערכת מאשר - המשתמש מסווג כצפר מקצועי.

9.2. תיאור זרימת התהליכים העיקריים

9.2.1. הרשמה והתחברות

בכדי להזדהות בפני המערכת ולהישמר כמשתמש רשום, על המשתמש להירשם למערכת. בעת כניסתו ליישומון, עליו להגיע למסך הSignUp (ניתן להגיע אליו ממסך הבית). כאשר המשתמש מגיע לעמוד הSign-up, הוא נרשם עם שם משתמש, שם פרטי, שם משפחה, סיסמה, מייל וסוג המשתמש (אם הוא חובב, מקצועי או מנהל). במידה ושם המשתמש לא קיים ואין משתמש נוסף עם אותה כתובת מייל ושם משתמש, המשתמש מתווסף לרשימת המשתמשים בשרת.

בפעם הבאה שהמשתמש ירצה להתחבר, הוא ימלא בעמוד הLog-in את שם המשתמש והסיסמה וכך יתחבר למערכת.

9.2.2. עדכון פרופיל

בכדי לעדכן נתונים בפרופיל המשתמש, על המשתמש להיכנס לעמוד הפרופיל דרך מסך הMain (אליו מגיע לאחר ההתחברות). המשתמש לוחץ במסך על עדכון פרופיל ובמסך שנפתח יוכל לשנות את הנתונים אותם הוא רוצה לשנות ולבסוף ילחץ על כפתור העדכון. הנתונים אותם הזין יתעדכנו בשרת.

9.2.3. זיהוי ציפור

בכדי לזהות ציפור המשתמש נכנס למסך Recognize Bird, שם המשתמש מעלה תמונה (ע"י צילום / העלאת תמונה קיימת מהמכשיר) ליישומון. היישומון שולח את התמונה מהמשתמש לאלגוריתם. האלגוריתם מקבל את התמונה, מעבד אותה ולבסוף – האלגוריתם מחזיר ליישומון את שם הציפור שזוהתה. היישומון מעביר את המשתמש למסך התוצאה, מבקש מהשרת נתונים אודות אותה הציפור, וכלל המידע אודות אותה הציפור מוצג למשתמש.

9.2.4. חיפוש ציפור

בכדי לחפש ציפור המשתמש נכנס למסך Search Bird, באותו המסך יש למשתמש אפשרות לחפש ציפור ע"י בחירה מרשימה, או בחיפוש שמי. כאשר נבחרת ציפור, היישומון שולח בקשה למידע אודות אותה הציפור ממסד הנתונים שעל הענן, ומקבל בתשובה את המידע אודות אותה הציפור. את תוצאת החיפוש ניתן לראות באותו העמוד של חיפוש הציפור.

9.2.5. מבט על מיקומי ציפורים

בכדי לצפות במיקומי ציפורים, המשתמש נכנס למסך Find on Map, באותו המסך המתממשק עם מערכת מפות, היישומון מקבל מהמכשיר החכם את מיקום המשתמש ומעדכן את המפה כשנקודת המרכז של המפה היא מיקום המשתמש. המשתמש יכול להגדיר את טווח הזמנים המעניין אותו בכדי לראות ציפורים באזור מתוך אפשרויות הפתוחות למשתמש. המפה מציגה למשתמש נקודות ציון של הציפורים שזוהו בסביבתו. בלחיצה על נקודת ציון שכזו, ניתן לראות איזו ציפור זוהתה וע"י איזה משתמש.

9.2.6. אישור תמונות לאימון האלגוריתם

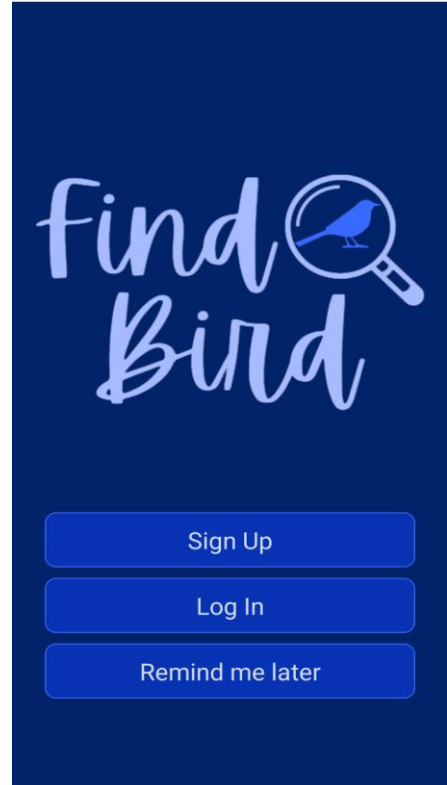
בכדי לאשר זיהויים יש צורך בקבלת תפקיד מנהל בלבד. למנהל מופיע במסך התפריט הראשי אפשרות להיכנס למסך Approve. במסך זה ניתן לגלול ולראות תמונות של ציפורים שזוהו על ידי האלגוריתם יחד עם שם הציפור שזוהתה בה, משפחת הציפורים והמשתמש שהעלה את התמונה. המנהל יכול לאשר את הזיהוי, ובכך לאפשר לאלגוריתם להתאמן גם עם תמונה זו, המנהל יכול לעדכן את הזיהוי (לשנות את שם הציפור שזוהתה) ולאחר מכן התמונה תישלח למאגר הנתונים והאלגוריתם יוכל להתאמן גם עם תמונה זו, וכמובן, המנהל יכול לבטל את הזיהוי ובכך להסיר את התמונה ממאגר הנתונים.

9.3. מסכי התהליכים העיקריים (ה-GUI)

9.3.1. מסך פתיחה

לפני כל התחברות ליישומון, המשתמש מגיע למסך זה על מנת להיכנס למערכת (ע"י הרשמה/ התחברות).

Home



איור 10-5 מסך פתיחה

9.3.2. מסך רישום

מסך הרישום מאפשר בתוכו למשתמש חדש להירשם למערכת.

← SignUp

SignUp

Username:

First Name:

Last Name:

Email:

Password:

Role: Hobby ▾

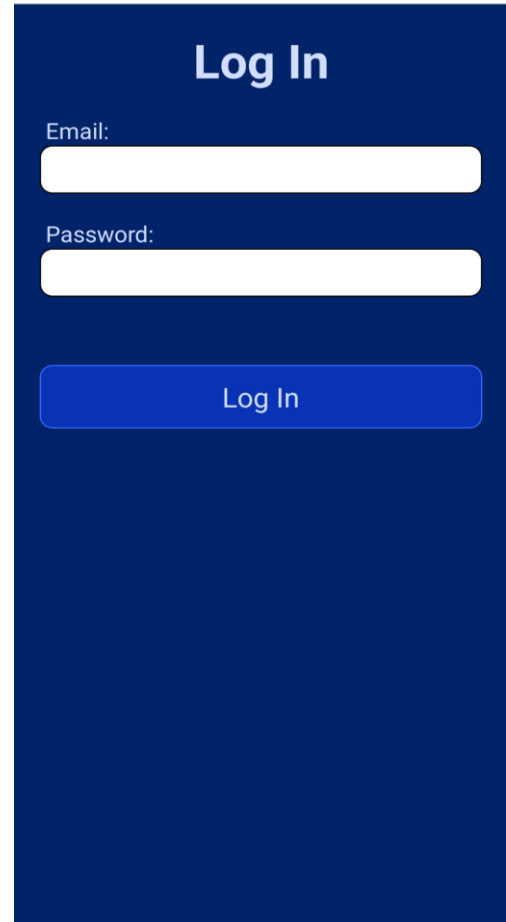
Sign Me Up

איור 10.6 מסך הרשמה

9.3.3. מסך התחברות

דרך מסך זה משתמשים רשומים יכולים להתחבר למערכת, במידה וההתחברות הצליחה, היישומון יעביר את המשתמש למסך התפריט.

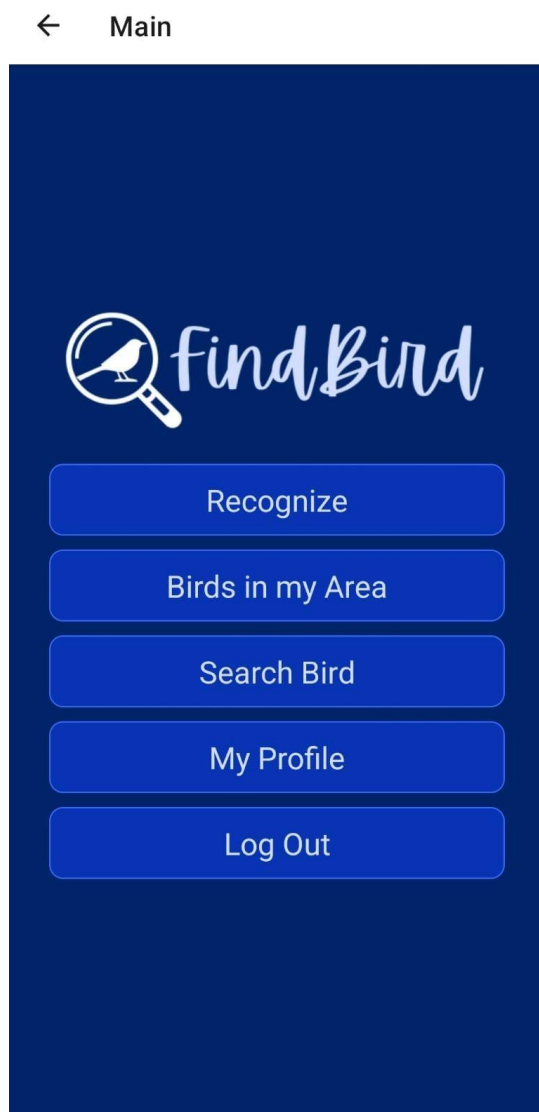
← Login

A screenshot of a mobile application's login screen. The screen has a dark blue background. At the top, there is a white back arrow and the text "Login". Below this, the text "Log In" is displayed in white. There are two white input fields: the first is labeled "Email:" and the second is labeled "Password:". Below the input fields is a blue button with the text "Log In" in white.

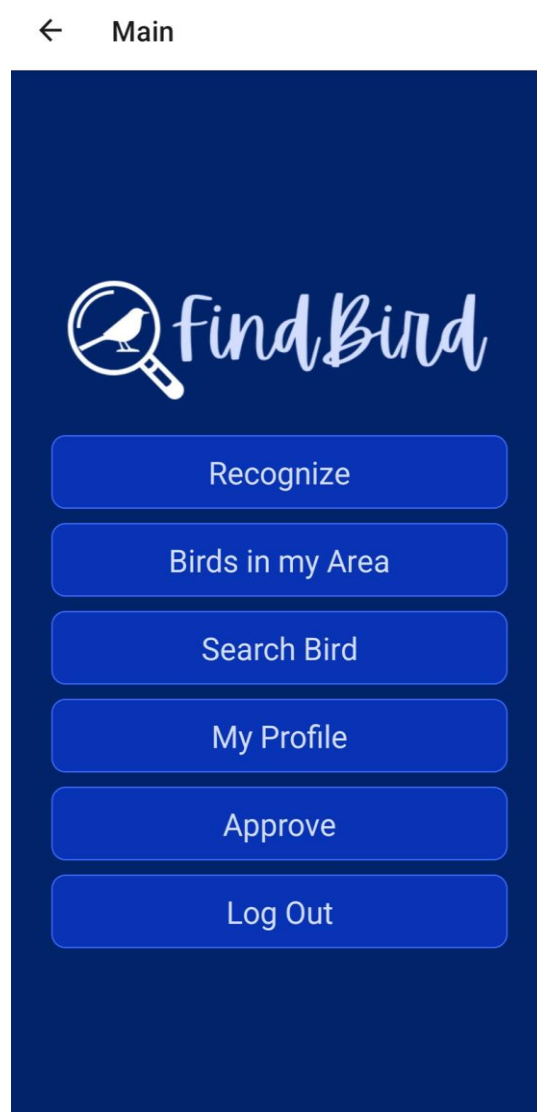
איור 10.7 מסך התחברות

9.3.4. תפריט ראשי

דרך מסך זה, המשתמש יוכל לעבור לשאר המסכים ביישומון. נשים לב שיש הבדל בין מסך התפריט הראשי של המנהל, למסך התפריט הראשי של משתמש רשום.



איור 10.99 מסך תפריט ראשי (משתמש רשום)



איור 10.89 מסך תפריט ראשי (מנהל)

9.3.5. חיפוש ציפור

במסך זה ניתן לחפש מידע על ציפור קיימת במערכת. ניתן לבחור את הציפור מתוך הרשימה (או לכתוב בתוך החיפוש בתיבה את שם הציפור), והציפור הנבחרת תוצג בהמשך המסך.

← Search

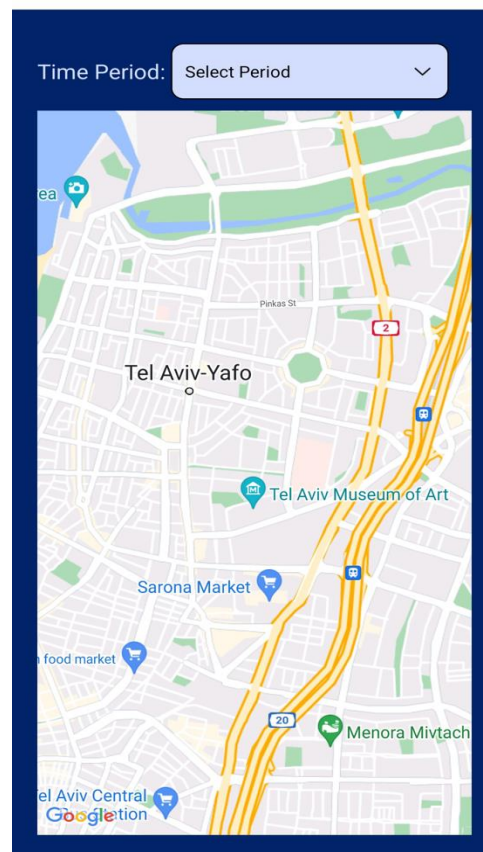


איור 10.9 מסך חיפוש ציפור

9.3.6. זיהוי ציפורים על מפה

המסך מקבל את האזור שהמשתמש נמצא בו ומראה את כלל הציפורים באזור שלו. ניתן לבחור את טווח הזמנים של הציפורים באזור בתיבה המופיעה למעלה.

← Area



איור 10.119 מסך זיהוי ציפורים על מפה

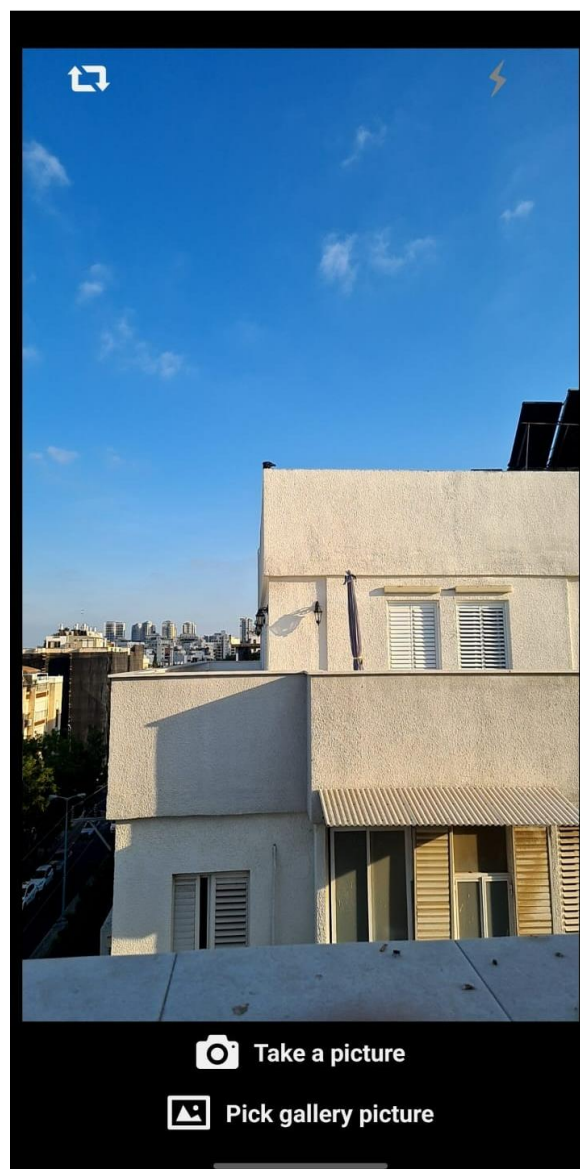
9.3.7. צפייה ועדכון פרטי משתמש

במסך הראשון המשתמש יוכל לצפות בפרטים המזהים שלו. בלחיצה על הכפתור "ערוך פרופיל" המשתמש מגיע למסך השני, שם יוכל לשנות את פרטיו.

9.3.8. זיהוי ציפור

במסך זה מופיעה מצלמה וכפתור גישה לספריית המדיה של המשתמש. המשתמש יכול להעלות תמונה או לצלם תמונה ולהעלות ליישומון. מסך זה שולח את התמונה לאלגוריתם וכאשר מגיעה התוצאה מהאלגוריתם, נשלח למסך התשובה.

← Recognize



איור 14.9 מסך זיהוי ציפור

9.3.9. מסך אימות זיהויים

במסך זה המנהל יכול לראות את כלל הציפורים שזוהו, לשנות את שמן (או את המשפחה אליה משתייכת הציפור), לאשר את הזיהוי או לבטל אותו. קיימת אפשרות נוספת של אישור כלל הציפורים בעמוד.



איור 10.15 מסך אימות זיהויים

9.4 ממשקים בין מודולים ותהליכים

9.4.1 התממשקות בין צד השרת לענן Heroku

בכל פניה מהיישומון בה המשתמש רוצה לשלוח או לשלף מידע מהשרת, עליו לגשת לשרת. ההתממשקות בין צד השרת לשרת בענן, הממוקמת על שרת הענן שהקמנו בHeroku, נעשת ע"י שימוש בכתובת URL לשרת ומשמות המפתחות הקיימים לכל הטבלאות.

בקשות ה-DELETE, GET, POST, UPDATE מתבצעות כבקשות http requests המתקבלות לכתובות של טבלאות המידע המעודכנים בשרתים ומשם המידע חוזר מהשרת בענן אל צד השרת של היישומון.

9.4.2 התממשקות בין צד השרת לשרת אחסון התמונות בS3

בשביל לשמור את התמונות על הענן, החלטנו להשתמש בשרת אחסון לתמונות של AWS הנקרא S3. בחרנו להשתמש בו כיוון שבניסיון עם שרת Heroku עליו אנו שומרים את שאר המידע, ראינו כי הפונקציונליות והמאפיינים של השרת לא מספיק חזקים ואינם יוכלו לעמוד בעומס התמונות מבלי לקרוס.

את S3 ניתן לחבר לצד השרת של היישומון ע"י שימוש ב-API מתאים.

9.5. הדגמת קישור מודולים וארכיטקטורה

מודלים וספריות חיצוניות - במהלך פיתוח תוצר Alphan של הפרויקט, נעזרנו בכלים הטכנולוגיים וספריות קיימות בReact-Native ובPython שמומשו בעבר והוטמעו (למשל מפות בReact-Native ומיקום המשתמש, מודלים קיימים וכו').

9.6. אלגוריתמים שמומשו במסגרת המערכת – אלגוריתם לזיהוי ציפורים

הקדמה

קלט: האלגוריתם מקבל תמונה המכילה ציפור מהמשתמש
פלט: האלגוריתם מחזיר ליישומן את הציפור שהוא זיהה בתמונה

באלגוריתם זה, האלגוריתם מקבל תמונה המכילה ציפור מהמשתמש וע"י פיענוח של התמונה בעזרת המודל בעל התוצאה הטובה ביותר מאימון האלגוריתם הקודם, האלגוריתם יחזיר תשובה ליישומן.

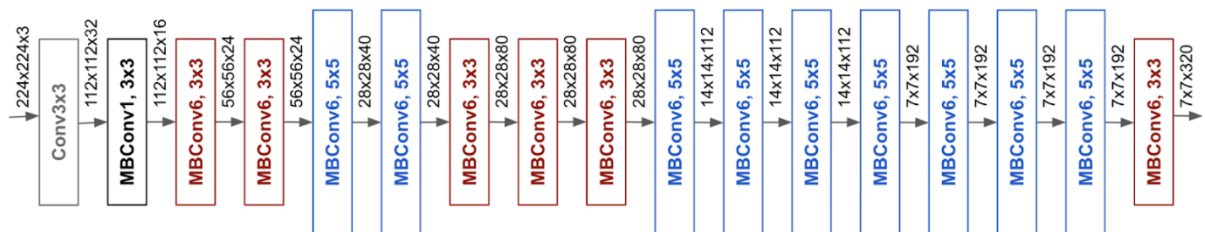
בחירת המודלים

בכדי לוודא שהאלגוריתם של היישומן יכיל את רשתות הנורונים המדויקות ביותר עבורו, החלטנו לעבוד מול מספר רשתות נורונים וראייה ממוחשבת את יעילותו. האלגוריתם ייאמן את הרשתות וישלח את התמונות שהמשתמשים מעלים לרשת המאומנת בעלת התוצאה הטובה ביותר.

EfficientNet B0

רשת קונבולוציה בעלת ארכיטקטורת רשת נורונים, המדרגת באופן אחיד את כל ממדי העומק/רוחב/רזולוציה באמצעות מקדם מורכב. בשונה מרשתות קונבולוציה אחרות המדרגות בצורה שרירותית, שיטת המדידה של רשת זו מאפשרת שימוש בקנה מידה אחיד של רוחב, עומק ורזולוציה עם מקדמים קבועים.

שיטת המדידה מוצדקת ע"י האינטואיציה שככל שתמונת הקלט גדולה יותר, אז הרשת זקוקה ליותר שכבות כדי להגדיל את השדה הקולט וליותר ערוצים בכדי ללכוד דפוסים עדינים.

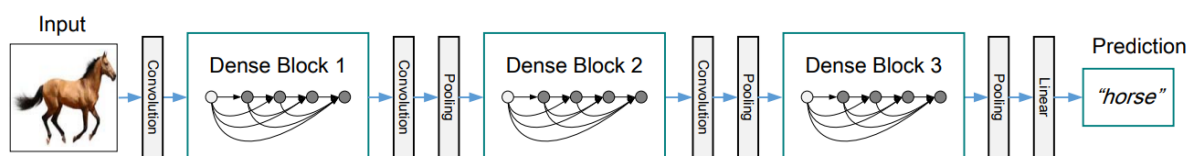


איור 10.16 ארכיטקטורה של EfficientNet B0

DenseNet201

ברשת קונבולוציה צפופה (DenseNet), כל שכבה מחוברת למספר שכבות נוספות בהמשך ע"י הערכה המתבצעת מראש לפי הציפיות (בניגוד ל"משוב" שנעשה לאחר המעשה). הן מקלות על בעיית הגרדיאנט הנעלם (בעיה שנוצרת כשמאמנים רשת נזרונים המבוססת ע"י למידת גרדיאנט ב-backpropagation. כל איטרציה מעדכנת את המשקולות ובמקרים מסוימים, הגרדיאנט עלול להיעלם, מה שלא יאפשר לרשת ללמוד כראוי – ובמקרי קיצון, יפסיק את הלמידה של הרשת), הן מחזקות את למידת הרשת, מעודדות שימוש חוזר במאפיינים קיימים ומפחיתים משמעותית את מספר הפרמטרים.

רשת DenseNet מבוססת על הרעיון שרשתות קונבולוציה יכולות ללמוד בצורה עמוקה יותר, מדויקת יותר ולאמן בצורה יעילה יותר אם יש להן חיבורים קצרים יותר בין השכבות לקלט ולפלט.

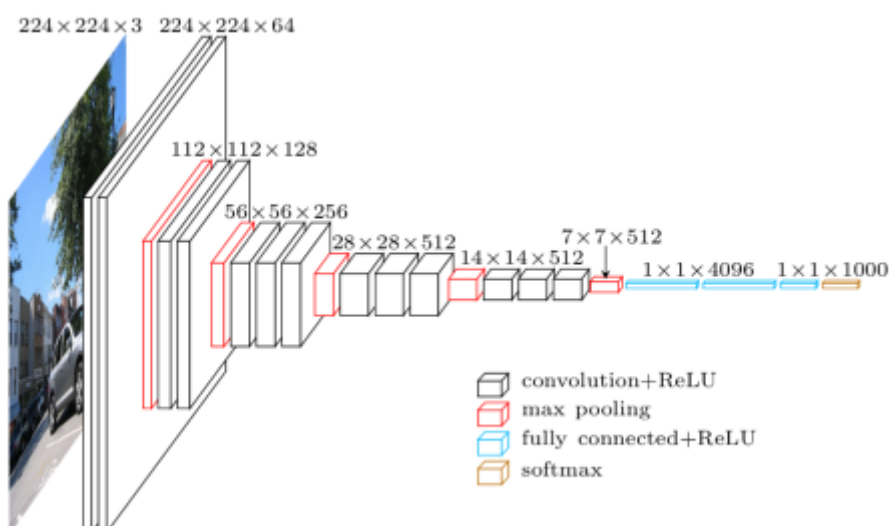


איור 10.17 ארכיטקטורה DenseNet

VGG16

רשת VGG הנה רשת קונבולוציה מבוססת רשת נזרונים (CNN), והיא נחשבת לרשת אימון מעולה. המאפיין המייחד שלה היא שבמקום להכיל מספר גדול של היפר-פרמטרים, הרשת מתרכזת בשימוש בשכבות קונבולוציה עם מסנן 3×3 עם stride 1 ותמיד משתמשת בריפוד של פילטרים מגודל 2×2 . לרשת יש שתי שכבות שהן מחוברות במלואן (Fully Connected).

ה-16 ברשת מתייחסת ל-16 השכבות שיש להן משקולות. הרשת הזו יחסית גדולה עם כבערך 138 מיליון פרמטרים.



איור 1-9 ארכיטקטורה VGG

1. איסוף תמונות למאגר התמונות ממנו האלגוריתם ילמד

הצורך הראשוני לאימון האלגוריתם הוא כמובן מאגר תמונות, דרך מאגר תמונות זה האלגוריתם יוכל ללמוד על הציפורים, להתאמן ולהיות מדויק יותר.

כאשר חיפשנו מאגר תמונות אמין בעל מספר תמונות שאינו מצומצם, ראינו כי קיים מאגר כזה באתר eBird בו יהיה ניתן להשתמש. לצורך השימוש, פנינו אליהם בבקשה להשתמש במאגר התמונות הקיים אצלם לצרכי לימוד בלבד. לאחר אימות נתונים אודות הפרויקט, אושר לנו השימוש במאגר התמונות שלהם.

יום ג', 22 במרץ, 22:27

Jenna Curtis <jc2585@cornell.edu>

אני, Lior

Hi Stav, thanks for confirming that! Your request has been approved and you should receive a separate confirmation email shortly.

,Best

Jenna

איוור 10.18 אישור לשימוש במאגר ע"י Jenna, נציגת eBird

במאגר שקיבלנו היו כבערך 9 ג'יגה בייט של תמונות, עם כבערך 1000 סוגים שונים של ציפורים מכל העולם (ובעיקר מאזור צפון אמריקה), כשלכל ציפור יש בין 30 ל-50 תמונות מהן ניתן ללמוד.

כיוון שבמשאבים הקיימים אצלנו, לא היה ניתן לאמן עם Database במשקל שכזה, בחרנו להשתמש ב-6 זנים של ציפורים שאת רובם ניתן לראות גם באזור ישראל, כשלבסוף Databases שלנו מכיל לפחות 300 תמונות בסיס. בנוסף, ע"י שימוש באוגמנטציות, הרחבנו את Databases ואת מספר התמונות הכולל שעל ידיו האלגוריתם התאמן.

2. ארגון וסידור התמונות

בכדי שנוכל להשתמש במאגר וגם לאמן בצורה היעילה ביותר, סידרנו את התמונות בהיררכיה הבאה: ראשית, את כלל התמונות הכנסו לאותה התיקיה – את התיקיה חילקנו לשלוש תתי-תיקיות: Train, Test, Validation. בעזרת קוד, חילקנו את התמונות הקיימות בצורה רנדומלית אך יחסית (שמכל ציפור יהיה אותו מספר יחסי של תמונות לכל תיקיה, כדי שהאימון יהיה יעיל יותר) ובכל תיקיה שכזו יש תתי-תיקיה עם שם הציפור (דהיינו, התיוג של הציפור לאחר מכן) והתמונות השייכות לאותה הציפור באותה קבוצה.

את תמונות המאגר חילקנו כך שמכל ציפור יבחרו רנדומלית לכל קבוצה כאשר 70% מהתמונות הגיעו לקבוצת האימון, 20% לקבוצת המבחן ו-10% לקבוצת האימות.

3. בניית Dataset

לצורך אימון האלגוריתם, היה צורך בבניית Dataset. אנו יצרנו את Datan ע"י קריאת התמונות מכל אחת מהתיקיות (עליהן הסברנו בסעיף הקודם) ורישום כל תמונה עם התיוג שלה. בנוסף, בשביל לאחד את כולן לאותו קובץ CSV במידה ונרצה להשתמש בו, כל לינק של תמונה נרשם יחד עם שם הציפור הקיימת בתמונה ואם זו תמונת אימון, מבחן או אימות.

4. בניית המודלים ואימונם

תחילה, יצרנו data ו-generator לכל קבוצה, בכדי שהמודלים יוכלו להשתמש בהם לאימון.

```
# Training data
train_datagen = ImageDataGenerator(rescale=1/255)
train_generator = train_datagen.flow_from_directory(PATH_TRAIN, target_size=(224,224),
                                                    batch_size= 30, class_mode='categorical')

# Test data
test_datagen = ImageDataGenerator(rescale=1/255)
test_generator = test_datagen.flow_from_directory(PATH_TEST, target_size=(224,224),
                                                  batch_size= 30, class_mode='categorical')

# Validation data
valid_datagen = ImageDataGenerator(rescale=1/255)
valid_generator = valid_datagen.flow_from_directory(PATH_VAL, target_size=(224,224),
                                                    batch_size= 30, class_mode='categorical')
```

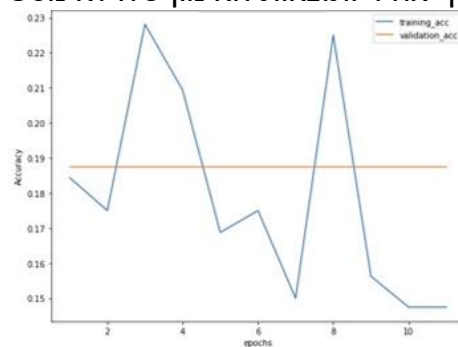
איור 10.199 מתוך הקוד, יצירת Datan וה-generator לאימון, מבחן ואימות.

לאחר מכן, כפי שהסברנו בתחילת הסעיף, השתמשנו במודלים בניהם אנו משווים: EfficientNetB0, VGG16, DenseNet201.

כל מודל קיבל את אותן קבוצות אימון, מבחן ואימות ולאחר מכן, קיבל את אותו מספר epoch ואימונים אותם עשה. בצורה זו אנו יכולים להשוות בין יכולות למידת האלגוריתמים בצורה הטובה ביותר ולראות שבהשוואה לאותם נתונים ובאותה נקודת פתיחה, מי האלגוריתם בעל התוצאה הטובה ביותר.

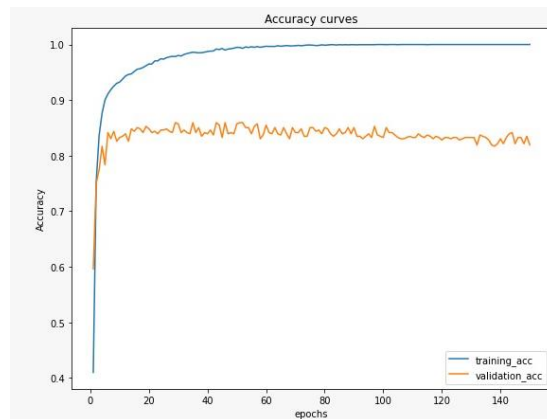
5. התוצאות

EfficientNet B0 – תוצאת האימון הכי פחות אמינה. ראינו כי באימון הרשת לא מתקבל גרף אחיד ותוצאות האימון שלו לא מספיק איכותיות.



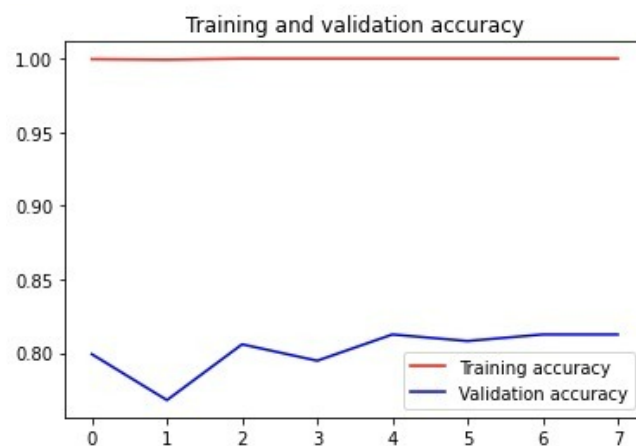
איור 10.209 תוצאות אימון EfficientNet

DenseNet201 – תוצאת האימון האמינה ביותר שקיבלנו, ראינו כי תוצאת האימון עומדת על 80% (Accuracy 0.815), מה שמאפשר להשתמש במודל זה לאימון.



איור 10.21 תוצאת האימון של DenseNet

VGG16 – ניתן לראות כי תוצאת הדיוק באימון מחשידה (עומדת על 100% הצלחה), אך תוצאת האימות יותר אמינה ועומדת על 81% (Accuracy 0.81), מה שמאפשר לנו להשתמש במודל זה לאימון.



איור 10.22 תוצאות האימון של VGG

ניתן לראות כי מבין התוצאות נכון לעכשיו האלגוריתם שני האלגוריתמים המובילים הנם DenseNet201 ו-VGG16, אך המודל שנותן את התוצאה הטובה ובמהימנות מדויקת יותר הנו המודל של DenseNet201 ולכן עד האימון הבא הוא יהיה המודל אשר יחזה את התוצאות.

6. חיזוי תמונה

האלגוריתם מקבל תמונה שהמשתמש שלח מהישומון. בשלב הראשון, המערכת טוענת את כלל המשקולות השמורות אצלה. לאחר מכן, המערכת בודקת מי המודל בעל התוצאה הטובה ביותר וטוענת את הרשת שלו עם המשקולות המעודכנים האחרונים. בשלב הבא, אנו נעדכן את התמונה לממדים בהם אנו מאמנים את התמונות ונעדכן את הגודל הBatch של הקלט. במקרה הזה, תמונה אחת. לאחר מכן, אנו נוודא כי התמונה בתווך הנכון מבחינת צבעים (כך שהערכים ביו 0 ל255), ע"י הרצה של הפונקציה `preprocess_input()` של Keras. ואז, אנו נבחר את המודל המאומן שלנו, נריץ אותו ונקבל את תוצאת הפלט. אם בתוצאת הפלט לא נראה כי התוצאה הראשונה נמוכה מ0.6, אנו נחזיר "OTHER", אחרת אנו נחזיר את הפלט ליישומון.

```
def get_best_model():
    mdl = [None, 0, 1]
    for res in saved_details:
        if (res[1] > mdl[1]) or (res[1] == mdl[1] and res[2] < mdl[2]):
            mdl = res
    return mdl[0]

def load_best_model(model_name):
    if model_name == 'DenseNet':
        t_model = tf.keras.models.load_model('myModels/DenseNet')
    elif model_name == 'EfficientNet':
        t_model = tf.keras.models.load_model('myModels/EfficientNet')
    else:
        t_model = tf.keras.models.load_model('myModels/VGG')
    return t_model

def get_img_predict(img_path):
    # Model choose and upload
    model = load_best_model(get_best_model())
    # Image
    img = image.load_image(img_path, target_size=(224,224))
    img_array = image.img_to_array(img)
    img_batch = np.expand_dims(img_array, axis=0)

    # Batch and preprocessing
    img_preprocessed = preprocess_images(img_batch)

    # run model
    prediction = model.predict(img_preprocessed)

    #prediction
    res = decode_predictions(prediction, top=1)[0]
    print(res)
    if res[1] < 0.6:
        return ("OTHER")
    return(res[0])
```

איור 10.23 מתוך הקוד, חיזוי תמונה

10. ממצאי ביצוע בדיקות מערכתיות

10.1. בדיקות שבוצעו בעבור פונקציות ותרשימים

| פונקציה / תרשים | צעד לביצוע | תוצאה רצויה |
|-------------------------|--|---|
| אלגוריתם לזיהוי ציפורים | קביעת אילוצים: קלט: תמונה המכילה ציפור הקיימת במאגר | האלגוריתם יחזיר פענוח נכון של הציפור בתמונה. |
| | קביעת אילוצים: קלט: תמונה ריקה (אין ציפור בפנים) | האלגוריתם יזהה שאין ציפור בתמונה והתוצאה שיחזיר תהיה ריקה. |
| | קביעת אילוצים: קלט: תמונה המכילה ציפור שאינה במאגר | האלגוריתם יזהה חוסר התאמה גבוהה עבור הציפורים הקיימות במאגר והתוצאה שיחזיר תהיה ריקה. |

טבלה 11.1 בדיקות שבוצעו בעבור היישומון

10.2. בדיקות שבוצעו בעבור היישומון

| מסך | צעד לביצוע | תוצאה רצויה |
|-----------------------|--|---|
| מסך הרשמה | כתיבת שם משתמש חדש, שם פרטי, שם משפחה, מייל חדש, סיסמה ותפקיד | המשתמש ירשם במערכת. |
| | כתיבת שם משתמש קיים, שם פרטי, שם משפחה, מייל חדש, סיסמה ותפקיד. | המסך יעדכן את המשתמש כי שם משתמש זה תפוס ועליו לבחור שם משתמש חדש. |
| | כתיבת שם משתמש חדש, שם פרטי, שם משפחה, מייל קיים, סיסמה ותפקיד | המסך יעדכן את המשתמש עם מייל אחר. |
| | אי כתיבת שם משתמש, כתיבת שם פרטי, שם משפחה, מייל חדש, סיסמה ותפקיד | המסך יעדכן את המשתמש כי לא ניתן להירשם ללא שם משתמש ועליו למלא את כלל הפרטים לפני שירשם |
| | אי מילוי כלל הנתונים בעמוד | המסך יעדכן את המשתמש כי עליו למלא את כלל הפרטים כדי להירשם. |
| מסך התחברות | כתיבת שם משתמש תקין וסיסמה תקינה | המשתמש יתחבר למערכת |
| | כתיבת שם משתמש תקין וסיסמה שגויה | המשתמש יקבל הודעה כי אחד הפרטים שמילא לא נכונים ויש למלא מחדש |
| | כתיבת שם משתמש לא תקין | המשתמש יקבל הודעה כי יש להזין את הסיסמה |
| חיפוש ציפור | בחירת שם מהשמות הנתונים | המסך יציג פרטים אודות אותה הציפור |
| | כתיבת שם של ציפור שאינו קיים במערכת | המסך לא יציג פרטים אודות הציפור וישלח הודעה "ציפור לא קיימת במערכת". |
| זיהוי ציפורים על המפה | בחירת טווח זמנים בו לא זוהו ציפורים | המערכת תראה מפה ללא זיהויים |
| | בחירת טווח זמנים בה זוהו ציפורים | המערכת תראה על המפה זיהויים באזור המשתמש. |
| עדכון פרטים | משתמש מוחק את כל פרטיו ולוחץ "עדכן" | המערכת תשלח הודעה כי לא ניתן לבצע פעולה זו ופרטיו הקודמים ישארו. |
| | המשתמש מעדכן את שם המשתמש לשם משתמש הקיים במערכת | המערכת לא תאפשר למשתמש לשנות את שם משתמש זה, פרטיו לא יתעדכנו. |
| | המשתמש מעדכן את שם המשתמש לשם משתמש חדש שאינו קיים במערכת | המערכת תאפשר למשתמש לעדכן את פרטיו |
| | המשתמש מעדכן את כתובת המייל לכתובת קיימת במערכת | המערכת לא תאפשר למשתמש לשנות את שם משתמש זה, פרטיו לא יתעדכנו. |
| | המשתמש מנסה לשלוח תמונה ריקה | המערכת תעדכן אותו כי לא זוהתה ציפור בתמונה. |
| זיהוי ציפור | המשתמש שולח תמונה של ציפור הקיימת במערכת | המערכת מעבירה את המשתמש למסך של מידה אודות הציפור שזוהתה באלגוריתם. |
| | המשתמש שולח תמונה של ציפור שלא קיימת במערכת. | המערכת תעדכן אותו כי הציפור לא קיימת במערכת. |
| | המנהל לוחץ על כפתור "אשר הכל" | כלל הציפורים שנמצאות בעמוד זה וממתינות לאישור, יתקבלו במערכת ויזוהו לפי הפרטים במערכת |
| אימות זיהויים | המנהל לוחץ על כפתור אשר עבור זיהוי ספציפי | המערכת מעדכנת את המאגר עם המידע אודות אותה ציפור. |

| | |
|--|---|
| המנהל לוחץ על כפתור מחק עבור זיהוי ספציפי | המערכת מוחקת את הציפור הקיימת מהמאגר והיא לא תופיע יותר. |
| המשתמש מעדכן את שם הציפור עבור זיהוי ספציפי ואז לוחץ אישור | המערכת תעדכן את המידע אודות אותה הציפור (המופיע במאגר המידע) והציפור תישמר. |

טבלה 11.2 בדיקות שבוצעו בעבור היישומון

11. סיכום ומסקנות

בכתיבת פרויקט זה, למדנו על המחסור במידע ובדרכים עבור צפרים מתחילים או חובבי ציפורים ללמוד על ציפורים ולזהות בזמן אמת את הציפורים בסביבתם. במסגרת התכנון נחשפנו לדרך בה ניתן לפתור את מצב זה בו אין מספיק מקורות טכנולוגיים דרכם החובב יכול לזהות ציפור ע"י צילום הציפור במכשיר הנייד, וע"י מימוש של טכנולוגיות הקיימות בשוק – איך ניתן לפתור את זה, בין אם פיתוח אלגוריתם או כתיבת יישומון. ניתן לראות כי ככל שיותר משתמשים ייקחו חלק וישתמשו ביישומון שפיתחנו, קהילת הצפרים יכולה להתרחב ולחשוף אנשים חדשים לתחביב (ואף למקצוע).

במהלך ביצוע הפרויקט, גם בחלק התכנוני וגם בחלק היישומי – הגענו למספר מסקנות חשובות:

11.1. שימוש במאגר תמונות רחב הינו קריטי לבניית האלגוריתם

במהלך הפרויקט נחשפנו לבעייתיות הקיימת כאשר אנו רוצים לאמן מודלים, אך מאגר התמונות ללמידה ממנו האלגוריתם לומד אינו גדול דיו.

במקרה שלנו, לפני שפנינו לeBird לקבלת המאגר המורחב, השתמשנו במאגר חינמי בו לא היו מספיק תמונות (לכל ציפור היו בין 3 ל-5 תמונות, מה שלא מספיק בשביל ללמוד את הציפור). ראינו כי אומנם האימונים מהירים, אך אין לנו מספיק תמונות לוודא את האימון והתוצאות במבחן ובוולידציה היו בהתחלה נמוכות מאוד.

החלפת המאגר והגדלתו מהווה מרכיב קריטי ומשמעותי להצלחת הפרויקט.

11.2. תכנון ראוי ועמידה בלוחות הזמנים הינם מרכיב קריטי להצלחת הפרויקט

במהלך הפרויקט למדנו על החשיבות של העמידה בלוחות הזמנים בשביל לצלוח את הפרויקט. ראינו שכאשר אנחנו עומדים ביעדים ובזמנים שהקצנו לכל תהליך, אנחנו רגועים יותר ועובדים בצורה יעילה יותר, האפקטיביות עולה והיכולת שלנו להתקדם בפרויקט טובה יותר, אך בחלקים מהפרויקט בהן היינו קרובים לדד-ליין שהקצבנו לעצמנו והיינו קרובים לאי-עמידה בזמנים, הלחץ והצורך להספיק כמה שיותר טרם הדד-ליין, השפיעה על ההנאה שלנו מכתיבת הפרויקט ועל התפוקה שלנו.

היכולת לתכנן כראוי את לוח הזמנים, בהתאם להיכרות של כל אחד מהמשתתפים בפרויקט ולעמוד בזמנים שהוגדרו, הנה חשובה ליכולת של הצדדים גם ללמוד אחד מהשני, גם לתקשורת וגם להצלחת הפרויקט.

11.3. בעת חיבור בין מרכיבים (צד שרת ללקוח, אלגוריתם לצד שרת וכו') עלולות לעלות בעיות שלא היו קיימות לפניכן

במהלך הפרויקט ראינו שלעיתים גם כאשר מרכיבים עמדו בעצמם בצורה טובה, כמו מסכים ובקשות מהשרת (מצד השרת לשרת ובחזרה), בעת חיבור בין המרכיבים יכולה להיווצר בעיה.

למשל, בחיבור הראשוני בין מסך הרישום למאגר הנתונים של הנרשמים, היה לנו קושי בחיבור, בקבלת המידע ובאינטגרציה בין הצדדים. אך לאחר שהבנו את המקור לקושי, היה לנו קל יותר להתגבר עליו מול חיבור בין מסכים למאגרים שונים גם כן.

העובדה שחישבנו מראש בלוח זמנים גם את "זמן הפציעות" והגדרנו זמן מראש עבור בעיות אינטגרציה, עזרה לנו לעמוד בזמני הפרויקט ולעבוד בצורה רגועה גם בעת בלת"מים (בלת"ם – בלתי מתוכנן).

11.4. תקשורת שותפת בין העובדים על הפרויקט חשובה להצלחתו

את העבודה על הפרויקט התחלנו בזמן מגיפת הקורונה. כחלק מ"תופעות הלוואי" של מגיפה עולמית, ראינו כי התקשורת הבין-אישית נעלמת, הנושא של פגישות סדירות ופרונטליות לא תמיד מתאפשר וכמובן, הקושי הנפשי של הרבה אנשים (ובפרט סטודנטים) לשלב תקשורת בין אישית עם עבודה ועם לימודים.

כאשר עובדים על פרויקט משותף, התקשורת בין הצדדים הוא תנאי הכרחי ביותר להצלחתו. חוסר תקשורת בין צדדים העובדים יחד יכול לגרום במקרים הקלים לאי הבנות, עבודה כפולה או נפילת נושאים בין הכיסאות, ובמקרים חמורים יותר גם לויכוחים, קצרים בתקשורת והפסקת עבודה יחד.

אחד הנושאים שהיו חשובים לכלל הצדדים בפרויקט היה שמירה על תקשורת שותפת, פגישות קבועות ודיווח על בסיס קבוע מה מצב כל אחד מהצדדים בין פגישה לפגישה. העובדה שהצלחנו לשמור על תקשורת טובה גם בתקופה שבה פחות יכולנו להיפגש פיזית, יחד עם שילוב טכנולוגיות כמו ZOOM בהן אנו יכולים לקיים פגישות כשכל אחד בבית שלו וגם, פגישות פרונטליות כשהמצב מאפשר (בעיקר לקראת סוף הפרויקט, כאשר המצב נרגע ואפשר לקיים פגישות פרונטליות) תרמו לתקשורת טובה בין משתתפי הפרויקט וכמובן תרם רבות להצלחתו.

12. הצעה לעבודת המשך

12.1. שיפור האלגוריתם

בהתאם לטכנולוגיה המתפתחת בתחום הראייה הממוחשבת ורשתות הנוירונים, נרצה שבעתיד המודל למציאת הציפור יכיל מודלים חדישים יותר שיוכלו להביא לתוצאות טובות יותר בפחות זמן פר-אימון.

12.2. פיתוח היישומון עבור חיות נוספות

הרעיון של זיהוי סוג, זן (או במקרה שלנו, ציפור) לפי מאפיינים בתמונה של ראייה ממוחשבת קיימת בשוק מספר רב של שנים. אנו מאמינים כי השימוש בטכנולוגיה הקיימת, באלגוריתם וביישומון שפיתחנו יאפשר למפתחים נוספים אפשרות להרחיב את האלגוריתם ולהשתמש בתשתית הקיימת שלנו לזיהוי זנים נוספים של חיות נוספות. למשל: גזעי כלבים, גזעי חתולים, סוגי דגים וכו'.

12.3. הוספת אפשרות של זיהוי ציפור לפי קול

בנוסף לטכנולוגיה הקיימת של זיהוי לפי תמונה בראייה ממוחשבת, קיימת גם טכנולוגיה של זיהוי לפי קול – NLP (Natural Language Processing), בה ניתן להשתמש וע"י שימוש באלגוריתמים ומודלים הקיימים בטכנולוגיה זו, יוכלו לזהות ציפורים לפי ציוצים.

<http://cs229.stanford.edu/proj2014/Aditya%20Bhandari,%20Ameya%20Joshi,%20Rohit%20Patki,%20Bird%20Species%20Identification%20from%20an%20Image.pdf>

<https://www.microsoft.com/en-us/research/wp-content/uploads/2017/07/Look-Closer-to-See-Better-Recurrent-Attention-Convolutional-Neural-Network-for-Fine-grained-Image-Recognition.pdf>

<https://ksiresearch.org/seke/seke20paper/paper122.pdf> [3]

https://www.researchgate.net/profile/Vani-Rajan/publication/339586575_CNN_BASED_CLASSIFIER_FOR_IDENTIFICATION_OF_CANINE_BREEDS/links/5e5a263892851cefa1cd98f2/CNN-BASED-CLASSIFIER-FOR-IDENTIFICATION-OF-CANINE-BREEDS.pdf

<https://merlin.allaboutbirds.org> [5]

<https://birdnet.cornell.edu> [6]

<https://www.birds.org.il/he> [7]

https://books.map.co.il/index.php?option=com_content&view=article&id=126:2011-08-24-09-41-24&catid=19&Itemid=15 [8]

https://www.tensorflow.org/tutorials/keras/save_and_load [9]

[EfficientNet | Papers With Code](#) [10]

<https://www.yesilscience.com/transfer-learning-densenet201> [11]

<https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c> [12]

<https://ebird.org> [13]

Aditya Bhandari, Ameya Joshi, Rohit Patki . Bird Species Identification from an Image. Department of Computer Science, Stanford University, Department of Electrical Engineering, Stanford University, Institute of Computational Mathematics and Engineering, Stanford University

Jianlong Fu , Heliang Zheng , Tao Mei. Look Closer to See Better: Recurrent Attention Convolutional Neural Network for Fine-grained Image Recognition. Microsoft Research, Beijing, China University of Science and Technology of China, Hefei, China

Lucas Pugliese Barros, Flavio Medeiros, Eduardo Moraes, Anderson Feitosa Junior. Analyzing the Performance of Apps Developed by using Cross-Platform and .Native Technologies. Informatics Coordination Federal Institute of Alagoas

Vani Rajan. CNN BASED CLASSIFIER FOR IDENTIFICATION OF CANINE BREEDS. Seventh International Conference on Advances in Information Technology & Networking (ICATN'20) At: Coimbatore

[5] Merlin Bird ID אתר

[6] BirdNET אתר

[7] אתר הצפרות הישראלי

[8] אתר לקסיקון מפה – המדריך השלם לציפורי אירופה וישראל

[9] חיזוי תמונה מאלגוריתם מאומן

[10] מידע על רשת ה-EfficientNet

[11] מידע על רשת DenseNet

[12] מידע על רשת VGG16

[13] eBird אתר

14. נספחים

14.1. פוסטר הפרויקט – מצורף כמסמך נפרד

14.2. נספח א' – סקירה ספרותית ראשונית

זיהוי זן הציפור הוא משימה קשה שעדיין עד היום, לעיתים ישנם ויכוחים על סיווג הזיהוי של הציפור.. זאת בעיה קשה שדוחפת את גבולות היכולות החזותיות עבור בני אדם ומחשבים כאחד. למרות שלהרבה ציפורים שונות יש את אותם החלקים (כמו מקור, כנפיים וכדומה), הזנים השונים של הציפורים מאובחנים בעזרת צורת חלקים אלו ובצבעם.

בנוסף, בדקנו את רמות הביצועים של אותם יישומונים שפותחו בטכנולוגיות השונות.

14.2.1. מאמר 1: Bird Species Identification from an Image [1]

המאמר פורסם בשנת 2011 ועוסק בזיהוי ציפורים בעזרת אלגוריתמי למידת מכונה שונים. במאגר ישנם 200 זני ציפורים שונים, 11,778 תמונות שונות שלהם. כך שבממוצע לכל ציפור ישנם 58 תמונות שונות לזיהוי.

המאמר מציג את האלגוריתמים שנבחנו ואת שיטת העבודה של כל אלגוריתם לניתוח התמונה, חיפוש המאפיינים המתאימים (למשל – צורת המקור, צבעי המקור, צורת הכנפיים, גודל וכו'). האלגוריתם שנבחנו הינם –

- סיווג בייסיאני נאיבי (Naïve Bayes) – שיטת סיווג בה ההנחה ה"נאיבית" אומרת כי אין תלות בין תכונות האובייקטים המסווגים כאשר סיווגם ידוע.
 - מכונת תמך וקטורי (Support Vector Machines) – טכניקת למידה מונחת לניתוח נתונים, סיווג ורגרסיה. המסווג נוצר על ידי מפריד ליניארי.
 - אלגוריתם שכן קרוב (K-nearest Neighbors) -אלגוריתם חסר פרמטרים, כאשר הלימוד בו מבוסס על המופעים בהם הפונקציה מקורבת מקומית.
 - ניתוח מפלה ליניארי (Linear Discriminant Analysis(LDA) – בשיטה זו נמצא שילוב ליניארי של תכונות המאפיין או המפריד בין מספר מחלקות או מקרים. לרוב בשימוש כמסווג ליניארי או לצורך הפחתת מימדים.
 - עצי החלטה (Decision Trees) – מודל חיזוי, הממפה תצפיות על פריט ויוצר מסקנות על ערך היעד של הפריט. במבנה של עצים אלה, עלים מייצגים סיווגים אפשריים וענפים מייצגים צירופים של תכונות אשר יובילו למחלקות הסיווג.
 - אלגוריתם יער אקראי (Random Forests) – שיטת למידה לסיווג ורגרסיה המבוססת על ריבוי עצי החלטה, בהן התוצאה מתקבלת משכלול עצי ההחלטה.
 - מסווג אחד מול השאר (One Vs Rest classifier with Logistic Regression) – בשיטה זו ישנו מסווג יחיד לכל מחלקה. כל מסווג בסיס מייצר ערך מספרי להחלטתו בכדי למנוע עמימות.
- לאחר ניתוח המאפיינים, העבירו את הנתונים והניתוחים שנעשו לתוך אלגוריתם למידת מכונה, אשר סיווג את הציפורים לפי המאפיינים שהתקבלו.

| Method | Training Accuracy | Testing Accuracy | Using certainty metric | Using PCA | Using Feature Selection | Using PCA + Feature Selection |
|---------------------|-------------------|------------------|------------------------|-----------|-------------------------|-------------------------------|
| Naive Bayes | 33.07 | 19.22 | | | | |
| KNN | 45.43 | 31.18 | | | | |
| Decision Trees | 99.83 | 24.35 | | | | |
| Random Forests | 99.39 | 33.58 | | | | |
| LDA | 63.56 | 45.44 | 46.73 | 47.81 | 47.7 | 47.38 |
| SVM | 50.67 | 43.91 | 48.15 | 48.74 | 49.11 | 46.93 |
| Logistic Regression | 84.42 | 51.61 | 52.42 | 53.31 | 51.02 | 53.65 |

טבלה 15.2.1 – טבלה מתוך מאמר [1], תוצאות האלגוריתמים הנבדקים במאמר

התוצאות הנתונות במאמר מצביעות על כך שלאחר חלוקה למאפיינים המשותפים לכלל הציפורים ולאחר שימוש באלגוריתמים הנבחרים, השימוש בשיטת סיווג של עץ החלטה היא השיטה בעלת אחוזי הדיוק הגבוהים ביותר במהלך האימון, עם אחוז דיוק של 99.83%, אך כאן ניתן לראות שבשיטה זאת מתקיים Over fitting משום שבמהלך המבחן ישנם רק 24.35%, אך בבחינה של השיטות – השיטה בעלת תוצאת הבחינה הגבוהה ביותר ללא - Over Fitting הינה רגרסיה לוגיסטית, עם אחוז דיוק של 53.65 במהלך יישומו יחד עם בחירת תכונות מסוימות (הורדו מספר תכונות שגם כן זיהוי שאינן עזרו לתוצאה, או אפילו פגועה בה).

ניתן לראות את אלגוריתם ה-LDA שאצלו במהלך המבחן מול קבוצת הולידציה הגיע לציון של 63.56% אך ברגע שהריצו אותו על קבוצה המבחן, הוא ירד בדיוק ברמה של 18% מה שיכול להעיד גם כן על התאמת יתר לקבוצת האימון.

אלגוריתמים אלו אינם הגיעו לתוצאה גבוהה כמו שציפו החוקרים, ואף במספר מקרים הגיעו למצב של התאמת יתר.

14.2.2. מאמר 2: Look Closer to See Better: Recurrent Attention Convolutional [2] Neural Network for Fine-grained Image Recognition

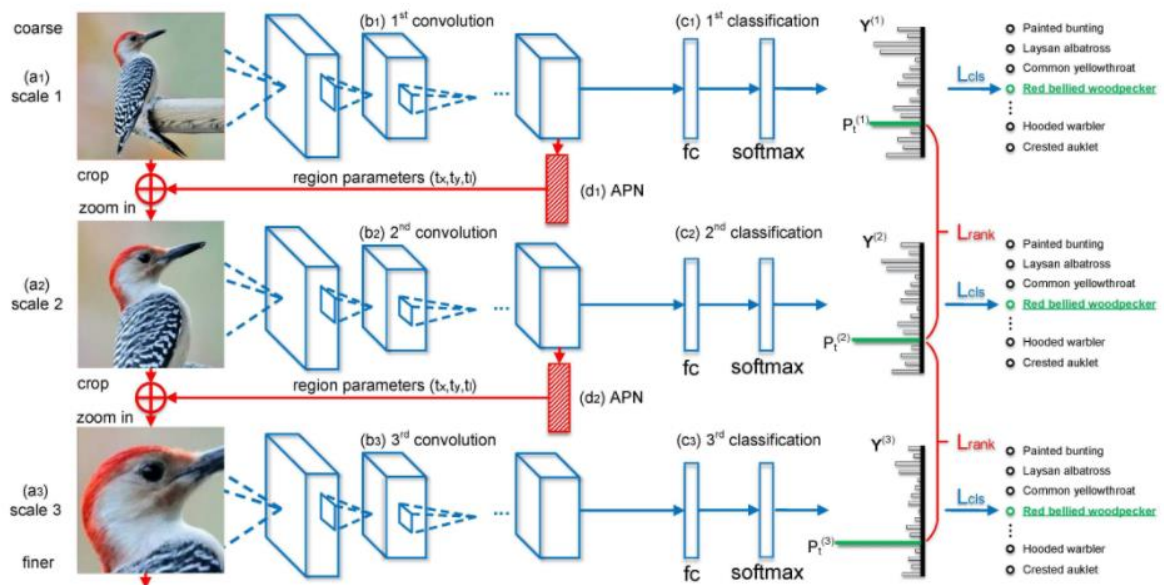
המאמר [2] פורסם בשנת 2017 ומציג את הדרך בה ניתן בעזרת רשת נוירונים לקבל תמונה אחת, ולהוציא ממנה מספר רב של נתונים על ידי רשת נוירונים מסוג RA-CNN.

רשת מסוג RA-CNN מקבלת קלט אחד, לאחר מכן מתבצע פיצול של התמונה לפי המאפיינים אותם הוא מזהה, הרשת מנתחת את אותו המאפיין ולאחר מכן הרשת מציגה את הניתוח הסופי של התמונה אותה קיבלה.

שיטת העבודה באלגוריתם זה הנה:

- קבלת תמונה.
- התמקדות על המאפיינים אותם הרשת תרצה לנתח.
- ניתוח כל מאפיין בנפרד.
- חיזוי התוצאה.

ניתן לראות באיור 15.2.2 את הדרך בה רשת הנוירונים מקבלת תמונה מלאה של ציפור ומתמקדת בראש הציפור על מנת לנתח את המאפיינים הרלוונטיים עבורה.



איור 14 – תמונה ממאמר [2], ניתוח תמונה ברשת נוירונים

14.2.3. מאמר 3: Analyzing the Performance of Apps Developed by using Cross-Platform and Native Technologies

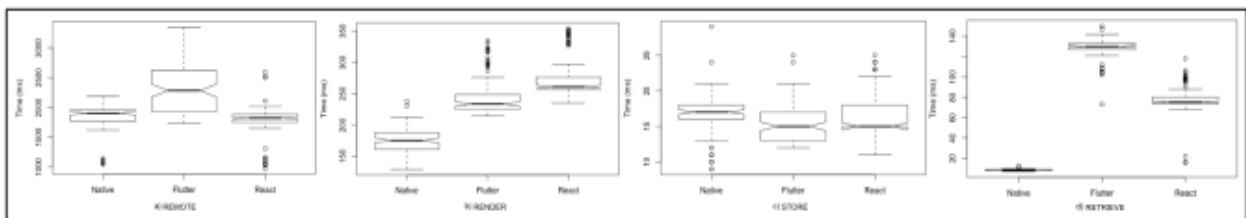
במאמר זה, בוחנים את רמת הביצועים של פיתוח אפליקציות בכל סביבת עבודה. המתחרים במאמר זה אלו הסביבות:

- "ריאקט נייטיב" (React Native) – סביבת פיתוח לאנדרואיד ו־IOS יחדיו, לא צריך לפתוח פרויקט מיעוד לכל סביבה וכותבים הכל בשפה משותפת. בסביבה זו כותבים בשפת JavaScript וספרייה זו פותחה על ידי חבר Facebook.
- "פלאטר" (Flutter) – סביבת פיתוח לאנדרואיד ו־IOS יחדיו, לא צריך לפתוח פרויקט מיעוד לכל סביבה וכותבים הכל בשפה משותפת. בסביבה זו כותבים בשפת Dart, שפה זו פותחה על ידי Google.
- פיתוח למערכת הפעלה מסוג IOS על ידי שימוש ב"סוויפט" (Swift) – סביבת פיתוח רק למכשירי IOS. הסביבה פותחה על ידי חברת Apple. בסביבה זו אי אפשר לפתח גם למכשירי Android.
- פיתוח למערכת הפעלה מסוג Android על ידי שימוש בשפת "ג'אווה" (Java) – סביבת פיתוח רק למכשירי Android, בשפה זו אי אפשר לכתוב פרויקט גם למכשירי IOS.

במאמר זה מחלקים את המבחנים לשני ייעודים שונים:

פיתוח לסביבת Android – בה קיימת השוואה בין React Native, Flutter, Java.

- החוקרים בחנו את זמני התגובה של היישומים אשר פותחו בדרכים שונות לצורך ביצוע אותן פעולות למכשירים בעלי מערכת הפעלה Android. תוצאות המסקנות הנן סטטיסטיות לפי הפעולות שבוצעו.
- החוקרים הגיעו למסקנה בה שכשמדובר על פונקציות הקשורות לפעולות במערכת ההפעלה, Java הייתה בעלת התוצאות הטובות ביותר, אך פיתוח בJava מותאם אך ורק למערכת הפעלה זו ואינה מותאמת למערכת הפעלה מסוג IOS.
- מסקנה נוספת אליה הגיעו החוקרים הייתה שכאשר בוחנים בקשות מול שרת או אחסון בתוך המכשיר הנייד, כלל היישומים מכלל סביבות הפיתוח הגיעו לאותו זמן תגובה.
- את תוצאות ההשוואות מהמאמר ניתן לראות באיור 15.2.3.



איור 15.2.314 – תוצאות מאמר [3]

14.2.4. מאמר 4: CNN BASED CLASSIFIER FOR IDENTIFICATION OF CANINE BREEDS [4]

במאמר זה מתבצע זיהוי גזע הכלב לפי תמונה. בעיית זיהוי גזע הכלב הנה בעיית זיהוי לא קלה, זאת משום קיימת כמות רבה של גזעי כלבים שונים, וגם – יש גזעים שונים של כלבים מאד דומים במבנה, צורה, גודל וצבע וקיימים הבדלים מעטים הלא נראים לעין בין גזעי כלבים, ולעיתים הרבה הבדלים ושנראים לעין.

- החוקרים השתמשו במאגר של 8351 תמונות של 133 סוגי גזעים שונים של כלבים. לאימון המודל החליטו החוקרים לחלק את המידע כך שלאימון ישנם 80% מכלל התמונות (6680 תמונות) לקבוצת האימות ישנם 10% מכלל התמונות (836 תמונות) ולקבוצת המבחן ישנם 10% מכלל התמונות. תמונות אלו של הכלבים נלקחו ממאגר הנתונים Flickr, ImageNet, Google Image. תמונות אלו הכילו 120 גזעים שונים של כלבים.
- ImageNet הוא מאגר מידע חזותי גדול המיועד לשימוש במחקר תוכנות לאובייקטים חזותיים, כמו לדוגמא, המאגר גם כן משמש לזיהוי סוגי כלבים. במאגר זה ישנו מעל 14 מיליון תמונות שהעבור ביאור כדי לציין אילו אובייקטים מופיעים בתמונות אלו. ישנם במאגר זה קטגוריות לדוגמא כמו "בלון" או "תות שדה" המוכבות מכמה מאות תמונות.
- החוקרים לקחו את מאגר התמונות והמירו את כל התמונות שהיו בגדלים וברזולוציות שונות לגודל של 224X224 כדי שיעבוד עם רשתות כמו VGG16, ResNe50, MobileNet, InceptionV3 וגם לגודל של 229X229 בשביל Xception.



איור 15.2.4 – תמונה ממאמר [4], תמונות לדוגמה של גזעי כלבים

- מאגר זה היה יחסית קטן משום שישנם 133 גזעים שונים של גזעים וכך יצא שלגזע בעל מספר התמונות הקטן ביותר היו 33 תמונות ולגזע בעל מספר התמונות הגדול ביותר היו 96 תמונות.
- לאחר בניית מאגר הנתונים, לקחו רשת CNN שאינה מאומנת כלל ואימנו אותה על מאגר התמונות. כתוצאה מאימון התמונות על מאגר זה החוקרים הגיעו לדיוק של 49.46% בקבוצת האימות ול-47.42% של דיוק בקבוצת המבחן.
- לאחר התוצאה היחסית הנמוכה שיצאה במקצה זה, החוקרים חילקו את הניסוי הבא לשני חלקים. בחלק הראשון, בנו החוקרים רשתות נוספות: VGG16, InceptionV3, Xception לא מאומנות והריצו אותם על המאגר השלם של ImageNet ששם משתמשים בהוצאת "פיצ'רים" של כל תמונה, שיטה שונה ממה שנעשה בעזרת רשת ה-CNN. לאחר מכן החוקרים לקחו את הרשתות שידעות עכשיו להוציא "פיצ'רים" והריצו אותה על קבוצת

האימון, הולידציה והמבחן של המאגר נתונים של החוקרים. את נתונים אלו החוקרים שמרו. החוקרים רק רצו את החלק של זיהוי החלקים הספציפים שניתן מרשתות אלו. כך עשו החוקרים לשלושת הרשתות. בחלק השני, החוקרים לקחו את המודל המאומן שהיה לכל רשת כזו ועל הפלט שלו החוקרים הוסיפו עוד רשת על מנת לזהות רק את גזע הכלב ולא את ה"פיצ'רים" שמקבלים מרשתות אלו. לאחר מכן החוקרים שוב אימנו את המודלים המשולבים שבנויים מזיהוי ה"פיצ'רים" ולאחר מכן בזיהוי סוג הכלב ובדקו את התוצאות.

| Pre-trained architecture ▾ | Validation ▾ | Test ▾ |
|----------------------------|--------------|--------|
| VGG16 | 54.76% | 59.89% |
| Xception | 83.71% | 84.97% |
| InceptionV3 | 76.86% | 79.78% |

טבלה 15.2.5 – טבלה ממאמר [4], השוואת מודלים

ניתן לראות שתוצאות אלו הגיעו לציונים הרבה יותר גבוהים מהרשתות מהניסוי הקודם וניתן לראות שהרשת בעלת הציון הגבוה ביותר הייתה הרשת Xception שהגיעה לציון של 83.71% בקבוצת האימון ו84.97% בקבוצת המבחן. החוקרים הגיעו למסקנה של רשת הבנויה בעזרת Xception מבוססת על רשת CNN עם מודלים שנעשו בעזרת Transfer learning בעלת הציון הגבוה ביותר כאשר מנסים לזהות גזעים של כלבים.

14.3. נספח ב' – סקר שוק והשוואת מתחרים

14.3.1. יישומון MerlinBirdID [5]

Merlin Bird ID הינו יישומון שמטרתו היא לזהות ציפורים לפי תמונה וקול. האלגוריתם של היישומון פותח ע"י Cornell Labs. היישומון זמין למכשירי Android ומכשירי IOS היישומון הינו חינמי. כדי להשתמש ביישומון זה המשתמש צריך להוריד את היישומון מ Google Play למכשירי Android ו Apple Store למכשירי IOS. לאחר הורדת היישומון, יש להירשם ואז אפשר להתחיל להקליט ציפורים או לצלם ציפורים ולקבל זיהוי של הציפורים לפי האלגוריתם של היישומון.

היישומון מתאים לכל הגילאים ומאוד קל לשימוש לכל המשתמשים בכל הגילאים. בנוסף כאשר האפליקציה מבצעת זיהוי של ציפור היא מביאה גם נתונים עליה.

יתרונות:

- מבצע זיהוי של ציפור גם לפי תמונה וגם לפי קול, ישנם שתי דרכים לבצע זיהוי של הציפור. כך שמשתמש יכול לבצע את הזיהוי בכמה דרכים.
- היישומון צבר לאורך השנים מוניטין והוא מאוד פופולארי בקרב חובבי צפרות.

חסרונות:

- נכון להיום אי אפשרות לחפש ציפורים לפי מיקום גאוגרפי ספציפי (אלא ע"י מדינה).
- נכון להיום לא ניתן למצוא מקומות קודמים שציפור מסוימת נראתה לאחרונה.

14.3.2. אתר BirdNet [6]

אתר BirdNET הינו פלטפורמה מחקרית- אזרחית, המאפשרת לזהות ציפורים ע"י צלילים. האלגוריתם לזיהוי מבוסס על למידת מכונה ורשת עצבית מלאכותית ופותח ע"י Cornell Labs. שתי הדרכים המרכזיות להשתמש בפלטפורמה זו היא ע"י העלאת קובץ שמע לאתר, באתר מתבצע הזיהוי ומוצגות תוצאות אפשריות לציפורים המשמיעות את צליל זה, או - ע"י הורדת היישומון, רישום והעלאת קבצי שמע/ הקלטה דרך המכשיר החכם.

יתרונות

- יוצרי הפלטפורמה מעדכנים באופן תדיר את האלגוריתם ואת מספר הציפורים שמזהה.
- ממשק קל לשימוש

חסרונות

- מזהה לפי שמע בלבד.
- הציפורים המוכרות לאפליקציה מאזור צפון אמריקה ואירופה.
- באתר הפלטפורמה, ניתן לקבל רק את שם הציפור ולא מידע עליה

14.3.3. אתר הצפרות הישראלי [7]

אתר הצפרות הישראלי שייך לחברה להגנת הטבע והוקם בסיוע של ציוני דרך, המחלקה לזואולוגיה באוניברסיטת תל אביב, קרן הדוכיפת ומשרד התיירות. בתוך אתר זה ניתן למצוא מידע על מרכזי צפרות בארץ, פעילויות, מאמרים, תצפיות וטיולים. בנוסף לכל אלו, באתר זה קיים מגדיר ציפורים. המגדיר הינו מוקד הידע המקיף ביותר על כלל ציפורי ישראל. בשביל למצוא ציפור ניתן לחפש אותה במגדיר ולקבל מידע על הביולוגיה שלה, זיהוי, תפוצה, תצפיות אחרונות ותמונות.

יתרונות:

- מציג מידע כולל ומקיף אודות הציפורים בישראל.
- מונגש בעברית.
- מאפשר לראות מיקומים של אותן הציפורים בארץ.

חסרונות:

- לא ניתן לזהות ציפור לפי תמונה או שמע.
- לא ניתן לחפש ציפורים לפי אזורים.

14.3.4. לקסיקון מפה: הציפורים - המדריך השלם לציפורי אירופה וישראל [8]

מגדיר ציפורים זה הינו ספר הכולל מידע מרוכז על מאות רבות של ציפורים ומציג מידע אודותן ואיורי צבע עליהן. הספר מכיל את כל המידע הדרוש לזיהוי ציפורים בכל עונות השנה, תיאורי מינים וציפורים, מקום החיות שלה, תפוצה, מאפיינים, מפות וכו'. הספר הינו מפעל משותף של הוצאת מפה, הוצאת הקיבוץ המאוחד ובשיתוף עם החברה להגנת הטבע, מרכז הצפרות הישראלי והמרכז הבינלאומי לחקר נדידת הציפורים בלטרון.

יתרונות:

- אינו מצריך חיבור לרשת
- מונגש בעברית.

חסרונות:

- הספר מכיל מעל 400 עמודים ובכדי לחפש ציפור צריך לחפש אותה לפי בצורה ידנית.
- לא ניתן לזהות ציפור בצורה "חכמה" לפי תמונה שמעלים, אלא לחפש מבין האיורים בצורה ידנית מהו האיור המתאים ביותר לאותה ציפור אותה רוצה לזהות מהתמונה.

14.3.5. השוואה

| אתר הצפרות הישראלי | לקסיקון מפה: הציפורים | יישומון Merlin Bird ID | אתר birdnet | יישומון שלנו |
|--|---|---|--|--|
| לא | לא | כן | לא | כן |
| כן | לא | כן | כן | כן |
| כן | לא | לא | לא | כן |
| כן | לא | לא ידוע | לא ידוע | לפי תמונות שהועלו ע"י מנהל מערכת ומשתמשים |
| לא | לא | בחירת אזור לפי מדינה, לא מציג מיקום מדויק. | לא | בחירת אזור לפי רדיוס |
| מחשב, מכשיר חכם | ספר מודפס | מכשיר חכם | מחשב, מכשיר חכם, בקרי Arduino ו- Raspberry pi | מכשיר חכם |
| ציפורים בישראל / נודדות מעל שמי ישראל | ציפורים בישראל עד נקודת הזמן בה המגדיר הודפס | עולמי | צפון אמריקה, אירופה | ציפורים שעוברות בשמי ישראל |
| עברית, אנגלית | עברית | אנגלית | אנגלית | אנגלית |
| חינם, ניתן לתרום | 169 ש"ח | חינם | חינם, ניתן לתרום | חינם |

טבלה 14.315.3.1 - השוואת מתחרים

14.4. נספח ג' – חלופות

14.4.1. חלופות טכנולוגיות למימוש

החלופות הטכנולוגיות אשר עתידות לממש את מטרות הפרויקט באופן הטוב ביותר, יהיו טכנולוגיות העוסקות בפיתוח יישומון למכשירים ניידים, תוך שימוש בבסיס נתונים לשמירת המידע, וכמוכן התמונות שיאפשרו לאמן את האלגוריתמים במהלך פיתוח הפרויקט.

14.4.2. אלגוריתם לזיהוי ציפור

אלגוריתם הזיהוי מהווה חלק משמעותי ובלתי נפרד מהפיתוח של המערכת. ללא זיהוי נכון של הציפורים - המערכת תפספס את המטרה המרכזית שלה. לכן חשוב לבחור בחלופה המתאימה ביותר לזיהוי המבוקש.

פיתוח באמצעות שימוש באלגוריתמים מבוססי תמונות התמונה

שימוש באלגוריתמים מבוססי שימוש מאפיינים בתוך התמונות עצמן – למשל שוני בצבעים, קווי מתאר, גדלים ויחסים וכו'. בדרך זו נוכל לזהות על ידי מאפיינים דומים ציפורים ומשפחות ציפורים.

פיתוח באמצעות שימוש באלגוריתמים מבוססי ראייה ממוחשבת ולמידה עמוקה

שימוש באלגוריתמים מבוססי ראייה ממוחשבת קיימים, שמטרתם תהיה לזהות את הציפור או את משפחת הציפורים של הציפור בתמונה.

נאמן את כלל האלגוריתמים על מאגר תמונות הציפורים, נבצע שינויים על התמונות (אוגמנטציות) כדוגמת חיתוך, הזזה, סיבוב, חידוד וטשטוש וכך נגדיל את מאגר האימון של האלגוריתם, נשווה בין התוצאות שהתקבלו בכל האלגוריתמים ונבחר בתוצאה המדויקת ביותר.

פיתוח באמצעות שילוב האלגוריתמים

שילוב בין אלגוריתמים מבוססי תמונות התמונה ואלגוריתמים מבוססי ראייה ממוחשבת ולמידה עמוקה. בדרך זו נוכל לאמן בדרכים שונות את המאגר ולשלב את היתרונות שבשתי השיטות.

השוואת חלופות

| שילוב האלגוריתמים | אלגוריתמים מבוססי ראייה ממוחשבת ולמידה עמוקה | אלגוריתמים מבוססי תכונות התמונה | |
|--|--|--|------------------|
| חיבור למאגר מידע, ספריות OpenCV, התקנת Python על המחשב. | | | רכיבים רלוונטיים |
| כתיבת אלגוריתם המתבסס גם על מודלים של תכונות תמונה וגם מודלים של למידה עמוקה. | כתיבת אלגוריתם המתבסס על מודלים של למידה עמוקה לצורך זיהוי התמונות. | כתיבת אלגוריתם המתבסס על מודלים של תכונות תמונה בו אנו נחליט מהם המאפיינים הדומים | אופן מימוש |
| מימוש מורכב | מימוש מורכב. | מימוש פחות מורכב. | רמת קושי למימוש |
| שילוב היתרונות שבאלגוריתמים מבוססי תכונות תמונה ובאלגוריתמים מבוססי ראייה ממוחשבת ולמידה עמוקה | התאמת האלגוריתם במדויק לצרכים שלנו, לפי הקריטריונים עליהם אנחנו רוצים להתבסס. במימוש והשוואה בין מספר מודלים התוצאה תהיה מדויקת יותר | התאמת האלגוריתם במדויק לצרכים שלנו, לפי הקריטריונים עליהם אנחנו רוצים להתבסס. מהיר יותר ממימוש של בלמידה עמוקה | יתרונות |
| זמן אימון ארוך מכל שיטה בנפרד. | דורש זמן אימון וזמן לקבלת תוצאה ארוך מזה של אלגוריתם מבוסס תכונות | האלגוריתם מתבסס בצורה יותר שטחית מעמוקה בזיהוי הציפור בתמונה | חסרונות |
| 8 | 7 | 6 | משקל התרומה |

טבלה 15.4.114.4 – השוואת חלופות אלגוריתם

החלופה הנבחרת

בהתאם לאפשרויות שהוצגו, בחרנו להשתמש בשילוב בין האלגוריתמים. בדרך זו, נוכל גם לקבל את היתרונות של שתי השיטות וגם להכיר יותר טוב את המודלים הקיימים, לממש ולשלב אותם בצורה הנכונה – וזו מטרה עצמית שהצבנו לעצמנו בפרויקט הגמר.

14.4.3. שפת פיתוח האלגוריתם

פיתוח אלגוריתם באמצעות Python

השימוש חינוכי, בPython קיימות ספריות רבות למתכנתים שרוצים לפתח בסביבה זו אלגוריתמים של ראייה ממוחשבת, כדוגמת Keras, Pytorch. בנוסף, קיימות ספריות כמו OpenCV לצורך עריכת תמונה. ממשק נוח בצורת קוד בלבד. מפתחים רבים בוחרים לממש ולפתח קוד בשפה זו.

פיתוח אלגוריתם באמצעות Matlab

השימוש חינוכי לסטודנטים, הממשק משתמש נוח ובצורת קוד בלבד. נוח למפתחים שאינם רוצים או יודעים לפתח בעזרת Python.

פיתוח בסביבת R

שימוש חינוכי, ממשק נוח בצורת קוד לבד. קיימות מגוון של ספריות לפיתוח האלגוריתם בסביבה זו. סטודנטים ובעלי מקצוע העוסקים בסטטיסטיקה וכלכלה מרבים להשתמש בה.

פיתוח אלגוריתם באמצעות C++

השימוש בשפה זו חינוכי, הפיתוח נעשה על ידי כתיבת קוד. בשפה זו ישנן ספריות רבות למתכנתים לבניית אלגוריתמי ראייה ממוחשבת, וכן גם ספריות כדוגמת OpenCV לעריכת התמונות. מפתחים רבים בוחרים להשתמש בה כיוון ששפה זו מהירה משפות עיליות אחרות, אך הכתיבה בשפה זו פחות אינטואיטיבית משפות אחרות (כדוגמת Python).

השוואת חלופות

| C++ | R | Matlab | Python | |
|--------------------|--------------------|---|------------------|--------------------|
| חינמי | חינמי | חינמי לסטודנטים בלבד, לאחר מכן בתשלום | חינמי | מחיר |
| V | V | יש לכתוב את הקוד בPython או בC++ ולהמיר לקובץ Matlab. | V | תמיכה בOpenCV |
| Visual Studio Code | R | Matlab | Jupyter Notebook | סביבת פיתוח אפשרית |
| ממוצעת | נוח | נוחות בינונית | נוח מאוד | נוחות לכתובת קוד |
| V | באופן חלקי | V | V | תמיכת הסביבה בGPU |
| בינוני למימוש | קל - בינוני למימוש | קל - בינוני למימוש | קל למימוש | רמת קושי למימוש |

טבלה 14.415.4.2 - השוואת חלופות דרכי פיתוח האלגוריתם

החלופה הנבחרת

בהתאם לאפשרויות שהוצגו, בחרנו להשתמש ב-Python, כיוון שהשימוש בשפה זו עבור פיתוח אלגוריתמי ראייה ממוחשבת נפוצה ורלוונטית. בנוסף, השפה נוחה עבורנו לשימוש (גם לאחר שהתנסינו בה), תומכת בOpenCV ובשימוש בGPU.

React Native

פלטפורמת React Native נכתבת ב-Java Script ופותחה ע"י חברת Facebook. פלטפורמה זו אינטואיטיבית לבניית ממשק משתמש עבור מפתחים ומאפשרת מגוון אפשרויות פיתוח. פלטפורמה זו מאפשרת לנו לפתח במקביל יישומון למכשירים חכמים בעלי מערכת הפעלה Android וגם למכשירים חכמים בעלי מערכת הפעלה iOS. דהיינו, פיתוח במקביל לשתי מערכות ההפעלה, מה שמקצר בהרבה את שלבי הפיתוח לעומת פיתוח יישומון נפרד לכל מערכת הפעלה.

יתרונות:

1. פיתוח עתידי למכשירים חכמים תומכי iOS, יתאפשר ללא פתיחת פרויקט חדש – אנו נבצע התאמות בקוד הקיים.
2. הפלטפורמה מודולרית ואינטואיטיבית.
3. קהילה המשתמשת בפלטפורמה זו רחבה וקיימת תמיכה רבה.

חסרונות

1. איתור השגיאות מסובך.
2. ספריה יחסית חדשה, קיימים עדכונים ושינויים רבים בסביבה בפרק זמן קצר.

פיתוח באמצעות Native Android

המשמעות בשימוש בפלטפורמה זו היא פיתוח יישומון תואם ספציפית לפלאפונים בעלי מערכת הפעלה מסוג Android. חברות רבות כדוגמת Samsung, LG, OnePlus משתמשות במערכת הפעלה זו, ולכן מערכת הפעלה זו מהווה נתח גדול משוק המכשירים הניידים החכמים. מערכת הפעלה זו תומכת בשיטת הקוד הפתוח ושפות התכנות לפיתוח ב-Native Android אלו Java, Kotlin.

במידה ונבחר לממש את האפליקציה בפלטפורמת Native Android, נוכל לאפשר למכשירים בעלי מערכת הפעלה של Android בגרסאות שונות להריץ את היישומון. אך שימוש בשיטה זו ימנע מאיתנו פיתוח למכשירים בעלי מערכת הפעלה iOS.

פיתוח באמצעות Native iOS

המשמעות בשימוש בפלטפורמה זו היא פיתוח יישומון תואם ספציפית לפלאפונים בעלי מערכת הפעלה מסוג iOS. החברה היחידה התומכת במערכת הפעלה זו היא Apple, אשר בעצמה מהווה נתח מכובד משוק המכשירים הניידים החכמים.

בכדי לפתח בסביבה זו, יש צורך להשתמש במחשב נייד של חברת Apple בלבד, וגם להשתמש בתוכנה xCode לפיתוח היישומון. השפות בהן ניתן לפתח ב-Native iOS אלו Objective C, Swift.

במידה ונבחר להשתמש בפלטפורמה זו, הדבר ימנע מאיתנו פיתוח למכשירים בעלי מערכת הפעלה Android.

פיתוח בפלטפורמת React Native

פלטפורמת React Native נכתבת ב-Java Script ופותחה ע"י חברת Facebook. פלטפורמה זו אינטואיטיבית לבניית ממשק משתמש עבור מפתחים ומאפשרת מגוון אפשרויות פיתוח.

פלטפורמה זו מאפשרת לנו לפתח במקביל יישומון למכשירים חכמים בעלי מערכת הפעלה Android וגם למכשירים חכמים בעלי מערכת הפעלה iOS. דהיינו, פיתוח במקביל לשתי מערכות ההפעלה, מה שמקצר בהרבה את שלבי הפיתוח לעומת פיתוח יישומון נפרד לכל מערכת הפעלה.

פיתוח בפלטפורמת Flutter

אפשרות זו גם כן פותח את האפשרות לפתח במקביל למכשירים בעלי מערכת הפעלה Android וגם למכשירים בעלי מערכת הפעלה iOS ביחד. באותו פיתוח כותבים לשניהם, מה שמקצר הרבה את שלבי הפיתוח לעומת פיתוח שני יישומונים שונים לכל מערכת הפעלה. Flutter נכתב בשפת Dart ופותחה על ידי חברת Google.

השוואת החלופות

| Flutter | React Native | Native iOS | Native Android | |
|---|--|----------------------------|---|-----------------------|
| Visual Studio Code | Visual Studio Code, Android Studio (emulator) | מחשב Mac סביבת פיתוח Xcode | Android Studio | סביבת פיתוח |
| v | V | X | V | תמיכה במכשירי Android |
| V | V | V | X | תמיכה במכשירי iOS |
| פיתוח לשני מערכות ההפעלה, Android, iOS באותו הפרויקט, תמיכה של Google | פיתוח לשתי מערכות הפעלה, Android, iOS באותו הפרויקט, תמיכה של Facebook, אפשרויות עיצוב מגוונות | תאימות למכשירי iOS. | תאימות למכשירי Android, סביבת קוד פתוחה | יתרונות |
| Dart | Java Script | Objective C, Swift | Java, Kotlin | שפות תכנות |
| מימוש פשוט | מימוש פשוט | מימוש מורכב ויקר | מימוש פשוט | רמת קושי למימוש |

טבלה 15.4.3 – השוואת חלופות SDK

החלופה הנבחרת

בהתאם לאפשרויות שהוצגו, בחרנו בפיתוח ב- React Native כיוון שJavaScript מוכר לנו (לעומת Dart ו-Swift למשל) ובנוסף, נוכל לפתח עבור iOS ו-Android יחדיו.

14.4.5. שימוש בשירותי שרת / ענן

Heroku

עבור צד השרת, נרצה שרת שיוכל להריץ את האלגוריתמים שלנו, לקבל בקשות מהלקוח בכל זמן נתון. ושלא נצטרך להתעסק עם חיבור לרשת של השרת, זמינות של הרשת אצלנו בכל רגע נתון. משום כך, נשתמש בשרת ענן שיוכל לעשות לנו זאת. אנו בחרנו להשתמש בשרת Heroku שיש לו התממשקות טובה עם Github שם אנו מנהלים את ניהול הגרסאות של הפרויקט.

יתרונות:

1. מאוד פשוט לשימוש, קל לבצע עדכון בסביבת הפיתוח.
2. מאובטח בעזרת טכנולוגיות אבטחת המידע המעודכנות ביותר.
3. בעזרת ה Containers של Heroku הנקראים dynos, לא צריך להתעסק בהקמת השרת, הם עושים זאת.

חסרונות:

1. איטי יותר מחלק מסביבות הענן בשוק.
2. לא ניתן לקנפג את כל הגדרות השרת, כלומר – אין יכולת לשלוט על השרת במלואו.

Amazon Web Service (AWS)

במקום שימוש Heroku נוכל להשתמש ב Amazon Web Services. באמצעות Java אנו נבנה את צד לקוח, בעזרת AWS אנו נרים את סביבת ענן, ובאמצעות MongoDB נעלה את הבסיס הנתונים.

יתרונות:

1. כוח חישוב חזק.
2. שליטה מלאה על המשאבים (שינוי, הורדה או הוספה).

חסרונות:

1. זמן רב לפריסת פרויקט.
2. ביצוע ידני של העלאת גרסאות חדשות.

החלופה הנבחרת

אנו נבחר ב Heroku בשל הסיבות שפורטו בסעיף זה

PostgreSQL

בסיס הנתונים איתנו אנחנו משתמשים כרגע הוא PostgreSQL שהוא בסיס נתונים רלציוני המפותח בתור פרויקט "קוד פתוח".

יתרונות:

1. מציג נתונים סטטיסטים על שימוש בשרת.
2. מאוד קל ללמוד להשתמש.
3. מתרגם בקשות מהקוד ל־SQL בקלות, מפשט את כתיבת שאילתות.
4. גישה לתוספים, דרכים נוספות לעבוד עם המידע.

חסרונות:

1. רוב תוכניות הקוד הפתוח אינן תומכות PostgreSQL.
2. איטי מחלק מבסיסי הנתונים הרלציונים בשוק.

MongoDB

באמצעות React Native או נבנה את צד לקוח, בעזרת Heroku או נרים את סביבת ענן, ובאמצעות MongoDB נעלה את בסיס הנתונים. בסיס נתונים זה יהיה בתצורת NoSQL. בסיס הוא כלי "קוד פתוח" אשר נועד לאחסון כמויות מידע ענקיות.

יתרונות:

1. מאפשר לנו התממשקות מאוד פשוטה עם כל פלטפורמה שנרצה לפנות אליו בעזרת שימוש בבקשות REST, מה שמאוד נוח לממש בעזרת JavaScript שעל זה בנוי הפיתוח בצד הלקוח.
2. משום שאנו משתמשים ב־NoSQL במקום ב־SQL מאוד נוח להכניס מידע בהרבה צורות שונות מה שמאוד מוגבל ב־SQL ששם חייבים לפי עמודות עם ערכים מסוימים.

חסרונות:

1. מאפשר לנו התממשקות מאוד פשוטה עם כל פלטפורמה שנרצה לפנות אליו בעזרת שימוש בבקשות REST, מה שמאוד נוח לממש בעזרת JavaScript שעל זה בנוי הפיתוח בצד הלקוח.
2. במידה ונצטרך, מאוד קשה לבצע יחסים בין טבלאות.

החלופה הנבחרת

אנו נבחר להשתמש ב־PostgreSQL בשל היתרונות שפורטו.

14.5. נספח ד' – תוכניות עבודה

| Task Name | Start | End | Aug-2021 | Sep-2021 | Oct-2021 | Nov-2021 | Dec-2021 | Jan-2022 | Feb-2022 | Mar-2022 | Apr-2022 | May-2022 | Jun-2022 | Jul-2022 |
|-----------------|------------|------------|----------|----------|-----------------|----------|----------------|----------|----------|--------------|----------|----------|----------|----------|
| SOW | 02/08/2021 | 07/10/2021 | SOW | | | | | | | | | | | |
| Progress Report | 08/10/2021 | 15/12/2021 | | | Progress Report | | | | | | | | | |
| Interim Report | 16/12/2021 | 30/03/2022 | | | | | Interim Report | | | | | | | |
| Final Report | 31/03/2022 | 17/07/2022 | | | | | | | | Final Report | | | | |

טבלה 15.5.1 – תאריכי הגשה לביצוע הפרויקט

| Task Name | Start | End | Aug-2021 | Sep-2021 | Oct-2021 | Nov-2021 | Dec-2021 | Jan-2022 | Feb-2022 | Mar-2022 | Apr-2022 | May-2022 | Jun-2022 | Jul-2022 |
|---|------------|------------|-----------------------------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| SOW | 02/08/2021 | 07/10/2021 | SOW (02-Aug-21 - 07-Oct-21) | | | | | | | | | | | |
| Progress Report | 08/10/2021 | 15/12/2021 | | | | | | | | | | | | |
| Study React Native | 10/10/2021 | 22/11/2021 | | | | | | | | | | | | |
| Looking for CV algorithms for birds recognizing | 18/10/2021 | 08/11/2021 | | | | | | | | | | | | |
| Starting writing the Progress Report | 01/11/2021 | 10/11/2021 | | | | | | | | | | | | |
| Develop first version for identifying algorithm | 08/11/2021 | 22/11/2021 | | | | | | | | | | | | |
| Training & Testing the first identifying algorithm | 15/11/2021 | 23/11/2021 | | | | | | | | | | | | |
| Finish the Progress Report | 22/11/2021 | 24/11/2021 | | | | | | | | | | | | |
| First Check Progress Report | 25/11/2021 | 09/12/2021 | | | | | | | | | | | | |
| Correction Progress Report | 09/12/2021 | 10/12/2021 | | | | | | | | | | | | |
| Second Check of Progress Report | 10/12/2021 | 13/12/2021 | | | | | | | | | | | | |
| Progress Report | 08/10/2021 | 15/12/2021 | | | | | | | | | | | | |
| Interim Report | 16/12/2021 | 20/03/2022 | | | | | | | | | | | | |
| Improving first version of identifying algorithm | 17/12/2021 | 02/01/2022 | | | | | | | | | | | | |
| Front-End Application screens (incomplete) | 09/01/2022 | 31/03/2022 | | | | | | | | | | | | |
| Back-End Application screens (incomplete) | 09/01/2022 | 31/03/2022 | | | | | | | | | | | | |
| 2nd training and testing the improved algorithm | 02/02/2022 | 16/02/2022 | | | | | | | | | | | | |
| Start to write the Interim report | 06/02/2022 | 01/03/2022 | | | | | | | | | | | | |
| Improving latest version of identifying algorithm | 17/02/2022 | 06/03/2022 | | | | | | | | | | | | |
| 3rd Training and testing of latest algorithm | 07/02/2022 | 06/03/2022 | | | | | | | | | | | | |
| Finish Interim Report | 01/03/2022 | 05/03/2022 | | | | | | | | | | | | |
| Check Interim Report | 06/03/2022 | 13/03/2022 | | | | | | | | | | | | |
| Correction Interim Report | 14/03/2022 | 15/03/2022 | | | | | | | | | | | | |
| Second check Interim Report | 15/03/2022 | 19/03/2022 | | | | | | | | | | | | |
| Final Report | 21/03/2022 | 17/07/2022 | | | | | | | | | | | | |
| Continue 3rd Training and testing of latest algorithm | 22/03/2022 | 27/03/2022 | | | | | | | | | | | | |
| Complete Application's Front-End | 22/03/2022 | 15/04/2022 | | | | | | | | | | | | |
| Complete Application's Back-End | 22/03/2022 | 15/04/2022 | | | | | | | | | | | | |
| Creating the last version of the algorithm | 27/03/2022 | 10/04/2022 | | | | | | | | | | | | |
| Start to write the Final report | 01/04/2022 | 05/04/2022 | | | | | | | | | | | | |
| Application's testing | 01/04/2022 | 01/05/2022 | | | | | | | | | | | | |
| Training & Testing the final version of the identifying algorithm | 10/04/2022 | 30/04/2022 | | | | | | | | | | | | |
| Connect between Front-End to Back-End | 16/04/2022 | 30/04/2022 | | | | | | | | | | | | |
| Connect the final algorithm to the application | 16/04/2022 | 10/05/2022 | | | | | | | | | | | | |
| Extend the data of known birds | 10/05/2022 | 20/05/2022 | | | | | | | | | | | | |
| Final Touches | 29/05/2022 | 06/06/2022 | | | | | | | | | | | | |
| "Stoppage time" - Fixes and delays | 06/06/2022 | 20/06/2022 | | | | | | | | | | | | |
| Finish Final Report | 20/06/2022 | 25/06/2022 | | | | | | | | | | | | |
| Check Final Report | 01/07/2022 | 07/07/2022 | | | | | | | | | | | | |
| Correction Final Report | 07/07/2022 | 09/07/2022 | | | | | | | | | | | | |
| Second check and submission of the final Report | 10/07/2022 | 17/07/2022 | | | | | | | | | | | | |

טבלה 15.5.2 – לוח זמנים מפורט לפרויקט