25.10.2021

מגישים: רועי מלמד 308224138   
 יורי מלניקוב 313741175  
  
 מנחה: פרופסור יזהר לבנר

דו"ח סיכום פרוייקט:

גילוי מאורעות אקוסטיים בסביבה רועשת

תוכן עניינים

* [פרק 1: מבוא](#_Toc85736409)
* [1.1 הקדמה 2](#_Toc85736410)
* [1.2 שיטת עבודה 3](#_Toc85736411)
* [פרק 2: הכנת המידע](#_Toc85736412)
* [2.1 תקציר 4](#_Toc85736413)
* [2.2 תיוג הקבצים 5](#_Toc85736414)
* [2.3 טעינת הקבצים 6](#_Toc85736415)
* [2.4 פרטי הקובץ וקביעת הפרמטרים 7](#_Toc85736416)
* [2.5 התמרת פורייה 8](#_Toc85736417)
* [2.6 סולם מל והצגה ספקטרלית 9](#_Toc85736418)
* [2.7 המידע שלנו 11](#_Toc85736419)
* [2.8 Band-pass filter 12](#_Toc85736420)
* [2.9 אוגמנטציות – 'מניפולציות על המידע' 13](#_Toc85736421)
* [2.10 Pickle files – עבודה מהירה יותר 15](#_Toc85736422)
* [2.11 חלוקה לקבוצות 16](#_Toc85736423)
* [פרק 3: בניית הרשת](#_Toc85736424)
* [3.1 תקציר 17](#_Toc85736425)
* [3.2 למידה מודרכת 18](#_Toc85736426)
* [3.3 רשתות נוירונים – Artificial neural network 19](#_Toc85736427)
* [3.3.1 רשתות נוירונים – הקדמה 19](#_Toc85736428)
* [3.3.2 רשתות נוירונים – שיטת עבודה 20](#_Toc85736429)
* [3.3.3 רשתות נוירונים – בדיקת מודל 22](#_Toc85736430)
* [3.3.4 רשתות נוירונים – תוצאות 23](#_Toc85736431)
* [3.4 רשתות קונבולוציה – Convolutional neural network 24](#_Toc85736432)
* [3.4.1 רשתות קונבולוציה – הקדמה 24](#_Toc85736433)
* [3.4.2 רשתות קונבולוציה – אופן פעולה 25](#_Toc85736434)
* [3.4.2 רשתות קונבולוציה – ארכיטקטורת המודל 27](#_Toc85736435)
* [3.4.3 רשתות קונבולוציה – תוצאות 30](#_Toc85736436)
* [פרק 4: מסקנות וסיכום 32](#_Toc85736437)
* [ביבליוגרפיה 33](#_Toc85736438)

# פרק 1: מבוא

## 1.1 הקדמה

הכיוון הראשוני היה לחפש נושא בתחום למידה עמוקה.

לאחר שהתבוננו באופציות הקיימות אצל פרופ' יזהר לבנר, ולאחר התייעצות איתו, בחרנו בפרוייקט זה מכיוון שהוא מתמקד בנושא שחיפשנו והמימוש הוא בעזרת רשת קונבולוציה, שפחות עסקנו בה בתואר.

בפרט, פרוייקט זה מתמקד בגילוי וזיהוי של קולות ציפורים(מאורעות אקוסטיים) בתוך הקלטות ארוכות ורועשות.

המטרה שלנו היא בניית רשת, לסיווג בינארי של קטעי קול, שתוכל לזהות האם בקטע קול של שניה, ממאגר מידע מסויים, קיימים קולות של ציפורים שונות, ושהרשת תהיה רובסטית לאורך הרבה קטעי קול עם רעשים מגוונים.

מאגר המידע שיש לנו לאימון הוא מאוד מצומצם ונדרשת התייחסות רחבה לכך שנדבר עליה בהמשך.  
המאגר שהכנו מכיל ספקטרוגרמות בסולם מל ואלו יהיו דוגמאות האימון לרשת שעיקר הלמידה שלה הוא בעיבוד תמונות והוצאת המאפיינים שלהם.

בעיה זו היא בעיית גילוי וזיהוי אך ניתן להתייחס אליה בנוסף גם כבעיית סיווג וזאת מכיוון שהמערכת לא לומדת לקבוע מתי הציפור התחילה לשיר ומתי הפסיקה, אלא, לומדת לסווג בינארית קטעי קול למקטעים שיש בהם ציפור ומקטעים שאין.

נרצה גם לראות ולהראות שרשת קונבולוציה אכן יותר טובה למשימה זו משאר סוגי הרשתות.

תהליך העבודה בין היתר דרש מאיתנו לבחון המון המון תוצאות של ריצות לאורך הזמן, בין אם זה בבוקר ובין אם זה לפני השינה בשעות לא כל כך נוחות.(אין תירוצים)

בהזדמנות זו נרצה להגיד תודה רבה לפרופ' יזהר לבנר שנפגש איתנו פעמים רבות בזום, שתמיד נענה לכל שאלה וגם על הידע הרחב שתרם לנו בין אם זה בפרוייקט ובין אם זה במהלך התואר.

## 1.2 שיטת עבודה

למידה עמוקה בפייתון בעזרת חבילת Keras של Tensorflow.

הלמידה מתבצעת על מקטעים מתוך קבצי קול גדולים בשיטת supervised learning כך שלכל קטע קול יש תיוג שקובע האם בקטע יש ציפור או לא.

תחילה נדרשנו לקחת 2 קבצי קול לא מתוייגים ולתייגם כדי לתרום לגודל מאגר המידע שכבר קיים.  
לשם תיוג הקבצים נעזרנו בתוכנה שנקראת Audacity שבה אפשר לייבא את קבצי הקול, להציג את הספקטרוגרמה שלהם ובעזרת האזנה והתבוננות בספקטרוגרמה, ניתן למצוא את קולות הציפורים ולתייגם.  
התיוגים נשמרים כקובץ טקסט עם תגיות זמן בצורה שבה יהיה ניתן לקרוא את הקובץ בפייתון ולבנות מערך תגיות (labels).

לשם הכנת המידע כתבנו קוד שאחראי לקרוא את קבצי הקול וקבצי התיוגים ולפרק את הקבצים לקטעים כך שלכל קטע קול יש תיוג שמתאים לו.

מימוש הרשת נעשה ע"י התקנה של חבילת Tensorflow שציינו את שמה, ובפרט Tensorflow-gpu אשר משתמשת בכרטיס המסך לשם הלמידה.  
השימוש בכרטיס מסך משפר את מהירות הלמידה משמעותית שכן בכרטיס המסך יש כמות עצומה של ליבות לעומת כמות הליבות שיש במעבד.  
הליבות של כרטיס אמנם לא יכולות לעשות פעולות חישוב מסובכות כמו ליבות של מעבד אבל מצד שני, בגלל הארכיטקטורה הפשוטה שלהן, אפשר לדחוס הרבה יותר ליבות כאלה על יחידת עיבוד אחת.

בכל ריצת אימון של המערכת נשמור את המודל הטוב ביותר של אותה ריצה בשונה מלשמור את המודל של סוף הריצה.  
לאחר מכן, נבדוק את תוצאות המודל ולפי התוצאות נסיק מסקנות ונשנה את הפרמטרים והארכיטקטורה של הרשת בהתאם.  
נחזור על תהליך זה מספר רב של פעמים עד שתתקבל רשת עם תוצאות מספיק טובות.

בנוסף נרצה לבחון עוד ארכיטקטורות שונות ומוכרות שמצאנו באינטרנט   
כגון resnet34 ו- VGG16 לשם השוואה מול הרשת שלנו.

# פרק 2: הכנת המידע

## 2.1 תקציר

יש לשים דגש על תהליך הכנת המידע שכן הוא חיוני מאוד ללמידה מוצלחת של הרשת.

אפשר להגיד שרוב זמן העבודה בפרוייקט הושקע בהכנת המידע (אם לא מחשיבים את זמני הריצה של הרשת).

כמו שכבר ציינו, המשימה הראשונה שלנו הייתה לתייג קבצים שזהו גם חלק נחוץ בהכנת המידע.

בהתחלה עבדנו על קובץ שמע אחד, עם קובץ התיוגים שמתאים לו, כדי לוודא שהכל עובד.

חילקנו את הקובץ לבלוקים של שניה עם חפיפות של חצי שניה.(הכמות תהיה בערך פי 2 עם חפיפה)

כל בלוק עבר התמרת פורייה כדי שנוכל לקבל את התצוגה הספקטרלית שלו.

למידת המערכת תהיה דומה מאוד ללמידה על תמונות כך שכל תמונה היא ספקטרוגרמת התדרים של הבלוק.

עבודה על בלוקים של תמונות מאפשרת לנו להשתמש באוגמנטציות ידועות על תמונות כדי להגדיל את כמות המידע ובכך לתרום ללמידת הרשת.

את מערך התגיות הגדלנו בהתאם להגדלת כמות התמונות.

לאחר שראינו שהכל עובד, ביצענו את התהליך על 17 קבצי קול וקבצי תיוג על מנת לקבל מאגר מידע של תמונות והתיוג שלהן, ואת המאגר שמרנו בקובץ אחד נפרד.

את התמונות חילקנו ל3 קבוצות: קבוצת אימון, קבוצת ולידציה וקבוצת מבחן  
ואותן הזנו לרשת שתלמד.

## 2.2 תיוג הקבצים

ישנו מאגר עם המון קבצי קול מהקלטות ארוכות כך שאורכו של כל קובץ הוא כחצי שעה.

הבעיה היא שאין הרבה קבצי תיוג, או יותר נכון, יש כמות מאוד קטנה של תיוגים.

כשהתחלנו את הפרוייקט התבקשנו לתרום למאגר בכך שכל אחד מאיתנו תייג קובץ אחד שלם.

תהליך זה לוקח כמה שעות, של האזנה והתבוננות בהצגה הספקטרלית של גלי הקול, על מנת למצוא את קולות הציפורים ולתייגם.

לשם כך, השתמשנו בתוכנה הנקראת “Audacity” שבעזרתה אפשר בקלות להציג את הספקטרוגרמה ולייצא קובץ טקסט שמכיל את תגיות הזמן בהן תייגנו את הציפורים.

כמונו, היו עוד אנשים שתייגו קבצים דומים אחרים כדי לתרום למאגר המידע, שהוא עדיין מועט.

Graphical user interface, application

Description automatically generated

## 2.3 טעינת הקבצים

המטרה שלנו בשלב זה היא לחלק את הקבצים למקטעים שבהמשך ישמשו כדוגמאות אימון למערכת.

תחילה נעבוד על קובץ בודד יחד עם קובץ התיוגים שלו כדי לראות שהחלוקה עובדת טוב.

שיטת החלוקה של קובץ הקול תהיה כזאת:

* נבדוק את תדירות הדגימה של הקובץ (כמות דגימות הקול בשניה)
* נחליט כמה דגימות נרצה בכל בלוק (אורך הזמן של הבלוק בשניות) – Frame length
* נקבע גודל קפיצה Hop length כדי ליצור חפיפה בין המקטעים על מנת לא לאבד מידע מהקצוות – תופעה שנקראת Spectral leakage.

מכיוון שידוע לנו אורכו של כל בלוק, כך גם ידוע לנו זמן התחלתו בפועל לפי קפיצות של אינדקסים ומכאן ששיטת החלוקה של קובץ התיוגים תהיה כזאת:

* עבור כל מקטע נעבור על חותמות הזמן של קובץ התיוגים עד שנמצא את חותמת הזמן שמתאימה לו
* אם אין כזאת, נכניס תיוג 0 ונמשיך למקטע הבא
* אם מצאנו חותמת זמן חופפת, נבדוק אם החפיפה מספיק גדולה(threshold) ואם היא מספיק גדולה נכניס תיוג 1, אחרת 0.
* את ה-threshold לגודל החפיפה נקבע לפי חותמת הזמן הקצרה ביותר בקובץ התיוגים, כלומר, המינימלית.  
  אם חלק מקול של ציפור גולש למקטע אחר, נבדוק האם החריגה גדולה יותר מה-threshold ואז נתייג 1 או שהחריגה הייתה מינורית ונתייג 0 בקטע שהציפור גלשה אליו.  
  ה-threshold המינימלי נקבע לכל קובץ בנפרד לפי קובץ התיוגים שלו.

## 2.4 פרטי הקובץ וקביעת הפרמטרים

פרטי הקובץ הבאים נכונים לגבי כל הקבצים שאיתם אנחנו עובדים:

* Sample rate: 44,100Hz
* Length: 00:29:59 \*hh:mm:ss
* Bitrate: 1411kbps
* File type: WAV

Frame length

אורך המסגרת(גודל המקטע) נקבע לפי כמות הדגימות שנרצה בכל מקטע.

בפרויקט זה רצינו לעבוד על מקטעים של שניה ולכן כל מקטע יהיה בעל אורך מסגרת של 44,100 דגימות לפי הנוסחה הבאה:

frame’s duration = 1

File’s sample rate = 44,100Hz

Frame length = File’s sample rate \* frame’s duration(in seconds)

Hop length

גודל צעד אשר קובע את כמות החפיפה שתהיה בין המסגרות בקובץ.

נהוג לקבוע גודל צעד שהוא חצי מגודל המסגרת וכך עשינו גם בפרויקט זה.

מכאן שגודל הצעד הוא 22,050 דגימות בין מסגרת למסגרת.

חפיפה זו נועדה לפתור את בעיית ה- Spectral leakage

Spectral leakage

בגלל האופן שבו התמרת פורייה עובדת, והאופן שבו המסננים עוברים על המקטע, אנחנו מאבדים מידע מקצוות הקטע.

לא ניתן להימנע מאיבוד מידע זה אך יש בידינו שיטות להקטנת האבידה.

מידע זה יכול להיות חיוני ולא נרצה לאבד אותו, ומכאן הרצון לביצוע חפיפה בין מקטעי הקול השכנים, שזו אחת השיטות להפחית את כמות המידע שמאבדים.

עוד שיטה היא לבחור מסגרות ארוכות יותר, אך במקרה שלנו לא נרצה מסגרות ארוכות כדי שלמערכת יהיה קל יותר ללמוד.

## 2.5 התמרת פורייה

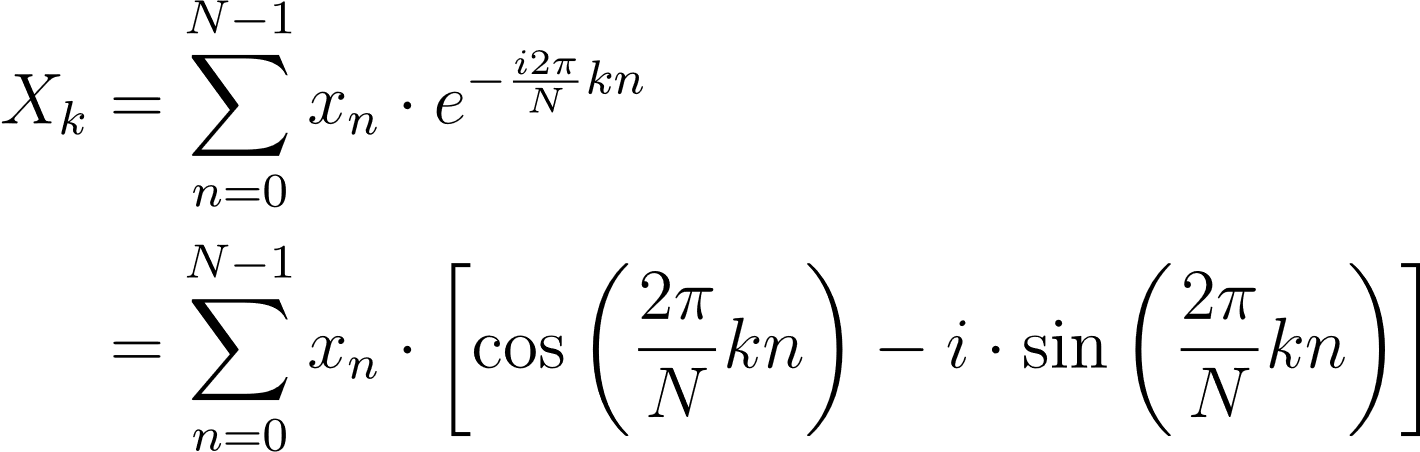
פירוק של כל פונקציה לצירוף של פונקציות הרמוניות פשוטות.

הרעיון המרכזי שעומד מאחורי התמרת פורייה הוא שניתן לייצג כל פונקציה מחזורית כסכום ממושקל של סינוסואידים פשוטים.

נשתמש בהתמרת פורייה כדי לפרק גל קול לתדריו השונים ועוצמתם, כלומר, לסינוסואידים המרכיבים אותו.

בעזרת פירוק זה יהיה ניתן להציג את ספקטרוגרמת התדרים של אותו הגל.

DFT – התמרת פורייה בזמן בדיד.   
התמרה זו היא ההתמרה היחידה בה לוקחים גל קול בדיד ומבצעים לו התמרה בזמן בדיד.  
הצורך שלנו בהתמרה זו הוא למצוא מרכיבי תדר של גל קול דגום במחשב לפי הנוסחה הבאה:



כך ש:

X(k) - התדר ה-k של האות הדגום

x(n) - דגימה במקום ה-n

N – כמות הדגימות

## 2.6 סולם מל והצגה ספקטרלית

ייצוג אות קולי באמצעות סולם מל היא אחת הדרכים המקובלות ביותר לניתוח מאפייני אותות קול. אחת הסיבות לכך היא מיצוי הספקטרום של האות על ידי מסננים המרווחים בדומה לדרך העיבוד של מערכת השמיעה האנושית.

הניתוח מתבצע באמצעות סידור מסננים בסקלה לוגריתמית על ספקטרום התדירויות של האות כשלאחר מכן מעבירים את האות דרך מסננים אלה ושומרים את המסננים כוקטור פרמטרים.

A picture containing text, clipart

Description automatically generatedנוסחת ההמרה, לסולם מל, לפי שאוגנסי:

כאשר f מסמל את התדר אותו רוצים להמיר לסולם מל.

ערכו של f נמדד ב-Hz, כלומר כמות המחזורים בשנייה שעושה גל התדר.

לאחר נוסחת ההמרה, נרצה להציג את התוצאות על ידי ספקטרוגרמה.

ספקטרוגרמה היא שיטת הצגה ויזואלית של ספקטרום התדרים המרכיבים את האות כך שציר ה-X של התמונה מייצג את הזמן וציר ה-Y מייצג את התדרים השונים.

Chart

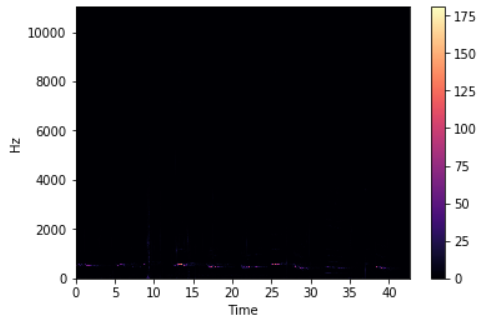
Description automatically generated with low confidence

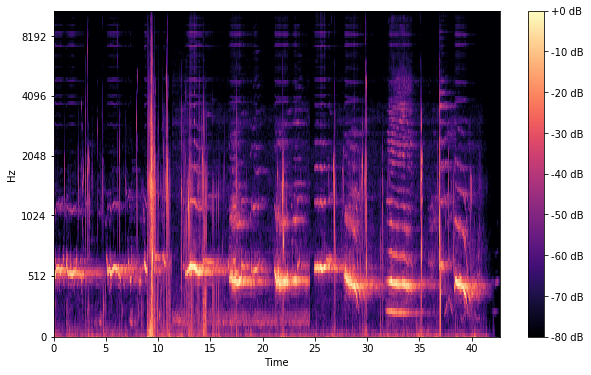
בפרויקט שלנו כל השלבים לעיל נעשים מאחורי הקלעים על ידי שימוש בחבילת Librosa בפייתון.

Librosa זאת ספרייה המיועדת לעבודה וניתוח של קבצי אודיו.

ספציפית למקרה שלנו, השתמשנו בפונקציה melspectrogram הנמצאת בתוך librosa.feature ואחראית להוצאה של מאפייני מל מגל הקול והפיכתם לספקטרוגרמה.

התוצאה שנקבל היא ספקטרוגרמה של תדרים בסולם מל והאמפליטודות(עוצמות) שלהם.

מכיוון שטווח העוצמות שבני אדם שומעים מוגבל מאוד, נמיר את העוצמות לדציבלים כדי לרווח את הספקטרוגרמה ולהקל על הנראות ע"י שימוש בפונקציה librosa.power\_to\_db:



כדי להציג את תוצאת הספקטרוגרמה נשתמש לבסוף בפונקציה specshow הנמצאת בתוך librosa.display.

## 2.7 המידע שלנו

ככל שהתמונה ברזולוציה גבוהה יותר (כלומר יש יותר מידע להצגת התמונה) ככה למערכת יהיה קל יותר לזהות מאפיינים בתמונה אך ככל שכמות המידע עולה ככה גם סיבוכיות הזמן וסיבוכיות המקום עולים.

לכן במקרה שלנו נרצה לשמור על מימדים קטנים של תמונה אך לא קטנים מידי כדי שהמערכת תצליח ללמוד.

לאחר מספר הרצות וניסויים הגענו למסקנה כי תמונה בגודל 175x70 מצד אחד אינה גדולה מדי ולכן לא תופסת יותר מדי זיכרון, ולא מכבידה על זמני הריצה ומצד שני מצליחה להביא את המערכת ללמידה מוצלחת.

רוחב התמונה נקבע לפי גודל מסגרת ההתמרה וגובה התמונה נקבע לפי כמות מסנני מל.

כזכור, עמודות התמונה/הספקטרוגרמה מייצגות נקודות זמן כך שבכל נקודת זמן נכנסת תוצאה של התמרה. ככל שגודל המסגרת גדול יותר, כך רוחב התמונה יקטן לפי החישוב הבא: (amount of samples / hop size) – 1.

כמות מסנני מל שבחרנו היא 90 כדי שלאחר פילטר מסוג Band-Pass נשאר עם 70 שורות (האופן בו מבוצע הפילטר אצלנו איננו קונבנציונלי ונרחיב על כך בהמשך).

בסכמה הבאה ניתן לראות את התהליך ואת כמות התמונות שמקבלים בסוף עבור כל קובץ בנפרד:

Graphical user interface

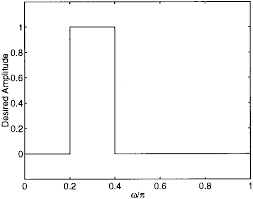
Description automatically generated with medium confidence

כך עושים עבור כל הקבצים ומחברים בין הדוגמאות שיצאו למערך אחד גדול של דוגמאות אימון.

## 2.8 Band-pass filter

ברוב הגדול של המקרים, קולות הציפורים ימצאו בטווח תדרים של- 1000-8000 Hz,   
לכן נרצה לבצע סינון מסוג band-pass כדי להפטר מהתדרים המיותרים למקרה שלנו ובכך להקטין את כמות השימוש בזיכרון, גודל התמונה, וזמן הלמידה.

תיאור גרפי של מסנן Bandpass אידאלי:



אנחנו מבצעים את הסינון על תמונות ומכיוון שהשורות של תמונות אלו מייצגות תדרים, כל מה שנדרש לעשות הוא פשוט להפטר ממספר שורות עליונות ותחתונות של התמונה ובכך לדמות סינון אמיתי של גל הקול עם הקטנה של התמונה.

כל ספקטרוגרמה, בלי יוצא מן הכלל, עוברת את הסינון הזה:

**Shape

Description automatically generated**

**Shape

Description automatically generated**

## 2.9 אוגמנטציות – 'מניפולציות על המידע'

A collage of a cat

Description automatically generated

כזכור, כמות המידע שעומדת לרשותינו קטנה מאוד ולכן יש לבצע פעולות, או "מניפולציות", כדי להגדיל את כמות המידע.

האוגמנטציות הגיעו מהצורך להגדיל את המידע, כאשר כמות המידע היא קטנה, בעזרת שינויים והפרעות שיכולים לקרות במציאות אך לא קרו בפועל.

אוגמנטציות מוכרות, לדוגמה, יכולות להיות:  
הזזת תמונה בציר ה-X\Y, מתיחת תמונה, הוספת רעש, סיבוב, היפוך וכו'

את המידע החדש נכניס כדוגמאות אימון חדשות עם התיוגים המתאימים להן.

Graphical user interface, application, PowerPoint

Description automatically generated

נתבונן בבעיה שלנו והיא סיווג קטעי קול, עפ"י תמונת הספקטרוגרמה שלהם, לקטעים שבהם יש ציפור וכאלו שאין.

לכן במקרה זה נרצה לבחור אוגמנטציות שנכונות לנו, או יותר נכון, הגיוניות בהשוואה לשינויים שיכולים לקרות בספקטרוגרמת התדרים של ההקלטות.

האוגמנטציות שבחרנו ליישם הן: Time shift, Pitch shift, Time stretch ו- Noise addition כך שבTime shift השתמשנו פעמיים ל2 הזזות שונות ואת הרעש מוסיפים כשכבה לרשת הקונבולוציה(נרחיב על כך בהמשך).

Time shift – הזזת המידע לאורך ציר ה-X.

ציר ה-X בספקטרוגרמות מייצג זמן ובחרנו לבצע הזזות של 0.2 ו- 0.4 שניות.

מן הסתם שקולות ציפורים לא תמיד יפלו על אותה נקודת זמן בספקטרוגרמה ולכן הזזה בזמן, במקרה זה, היא השינוי הנפוץ ביותר בין תמונה לתמונה ומכאן הסיבה שביצענו אוגמנטציה זו פעמיים.

אם נתבונן במקטעים שחילקנו, נוכל לראות שבעזרת מחיקה פשוטה של 0.2 השניות הראשונות של קובץ הקול, וחלוקה מחדש למקטעים לפי השיטה הרגילה, אנו נקבל מקטעים שהם זהים למקוריים רק שכל תמונה תהיה בהזזה של 0.2 שניות שמאלה(כך גם לגבי 0.4) בצורה הבאה:

Graphical user interface

Description automatically generated

Pitch shift – הזזת המידע לאורך ציר ה-Y, שמייצג את התדירות.

אוגמנטציה זו נובעת מהסיבה שציפורים לא תמיד מצייצות באותו התדר במדוייק, לפעמים קצת יותר גב וה ולפעמים קצת יותר נמוך כך:

Chart

Description automatically generated with low confidence

Chart

Description automatically generated with low confidence

Time stretch – מתיחת המידע לאורך ציר ה-X.

Chart

Description automatically generated with medium confidenceChart

Description automatically generated with medium confidenceפעולה זו מדמה ציוץ ארוך יותר של הציפור, מקרה שיכול לקרות במציאות.

## 2.10 Pickle files – עבודה מהירה יותר

תהליך הכנת המידע שכולל טעינת קבצים לזיכרון, התמרות פורייה, אוגמנטציות, הכנת מערך תגיות וכו'.. יכול לקחת מספר שעות ולכן נרצה לשמור את המידע המעובד, וזאת נעשה ע"י שמירה לקבצי pickle.

קבצי pickle הם קבצים השומרים בתוכם אובייקט פייתון(יכול להיות מחלקה שאנחנו יצרנו) וקישור למקום בו האובייקט נוצר.

טעינת הקובץ לוקחת מספר שניות ולעתים דקות לעומת כמות השעות שלוקח להכין אותו.

## 2.11 חלוקה לקבוצות

עבור למידה נכונה של הרשת נחלק את המידע שלנו ל 3 קבוצות:

Train set

על קבוצה זו הרשת שלנו לומדת לזהות מאפיינים ותבניות החבויים במידע.

כך היו עושה במשך מספר חזרות(epochs) ובכל פעם מעדכנת את משקלות הרשת.

לרוב עבדנו על training set של 70% מסך כל המידע.

Validation set

קבוצה זו נועדה כדי לבדוק את איכות הלמידה בזמן האימון, כלומר, בכל חזרה תתבצע בדיקה מול קבוצת הולידציה כדי לאשר שהרשת מצליחה ללמוד.

בסוף האימון נוכל להשוות גרפים שמעידים על התקדמות הלמידה מול קבוצת האימון לעומת התקדמות הלמידה מול קבוצת הולידציה ובכך לזהות מצבים בעייתיים כמו overfitting או שבכלל המערכת לומדת רק את קבוצת האימון בלי יכולת להשליך את הידע על שאר העולם.

לרוב ה- validation set היה 15% מסך המידע.

Test set

קבוצה זו אמורה לדמות מצב בו אנחנו בודקים את הרשת המאומנת על מידע שהיא לא התאמנה עליו לעולם.

בדיקה זו תתבצע בסוף האימון ובסוף כל החזרות כדי לבחון את הרשת.

לרוב ה- test set היה 15% מסך המידע.

בפועל החלוקה מתבצעת בפייתון בעזרת פונקציות פיצול מידע של חבילת sklearn כך שהפונקציות מחלקות את המידע לפי האחוזים הרצויים עם התחשבות בתגיות, זאת על מנת שהקבוצות יהיו מגוונות.

# פרק 3: בניית הרשת

## 3.1 תקציר

בתהליך זה, עיקר העבודה הוא הרצת ניסויים רבים ככל הניתן על מנת להגיע לדיוק גבוה ככל האפשר.

את הרשת נבנה בשיטת supervised learning שבה הרשת מקבלת דוגמאות אימון יחד עם תיוגים מתאימים ובכך היא לומדת.  
יותר ספציפי למקרה שלנו, התיוגים צריכים להיות בינאריים כך שכל תיוג קובע אם בדוגמת האימון קיימת ציפור או לא.

מלכתחילה ידענו שלבעיה זו רצוי להשתמש ברשת קונבולוציה אך כדי להעלות לעצמינו את הבטחון, תחילה, בדקנו האם בכלל ניתן לקבל תוצאות טובות על ידי שימוש ברשת נוירונים.

ספויילר: לא.

לאחר שראינו שהתוצאות לא מבטיחות ולאחר בדיקה נוספת של K-fold cross validation, הצלחנו להעלות לעצמינו את הביטחון ועברנו לבנייה של רשת הקונבולוציה.

Diagram

Description automatically generated

## 3.2 למידה מודרכת

הגדרתה של למידה מודרכת היא למידה על דוגמאות אימון פתורות כך שלכל דוגמה יש תיוג של הפתרון המתאים לה.

צורה זו של למידה מאפשרת למערכת ללמוד בעיות חישוביות של העולם האמיתי ובכלל, בעיות שיש לנו לגביהן ידע וניסיון.

נציין שוב שאוסף הדוגמאות מחולק לשלוש קבוצות:

* + קבוצת אימון – לרוב 70% מאוסף הדוגמאות.
  + קבוצת ולידציה – לרוב 15% מאוסף הדוגמאות.
  + קבוצת מבחן – לרוב 15% מאוסף הדוגמאות.

בשונה מלמידה לא מודרכת שבה אין תיוגים ואין חלוקה לקבוצות.

תיוגים אלו עוזרים למערכת להבין בסוף כל החלטה אם היא טעתה או צדקה ובהתאם לכך לעדכן את הפרמטרים בעזרת פונקציות המחיר וההפסד.

פונקציית ההפסד היא פונקציה הממפה מאורע או ערכים של משתנה אחד או יותר למספר ממשי המייצג "הפסד" של אותו המאורע, או במילים פשוטות יותר, בודקת כמה החלטה מסוימת הרחיקה את הרשת מפתרון או כמה קירבה אותה לפתרון.

בדומה לפונקציית ההפסד של קבוצת האימון, קיימת גם פונקציית הפסד לקבוצת הולידציה כדי שנוכל להשוות ביניהן בנוסף להשוואה בין הדיוק של 2 הקבוצות.

כיוון שמטרתנו היא לאמן מערכת, שתוכל לזהות ציפורים בקטעי קול, נרצה להשתמש בלמידה מודרכת מכמה סיבות:

* נוכל להשיג את מטרתינו מהר יותר
* יש בידינו ידע רחב על הבעיה ולכן יותר קל לכוון את המערכת בעצמינו

## 3.3 רשתות נוירונים – Artificial neural network

### 3.3.1 רשתות נוירונים – הקדמה

רעיון בסיסי: שיטת מידול שמדמה את אופי פעולתו של המוח.

רשתות אלו בנויות משכבות של קודקודים: שכבת כניסה (input layer), שכבות חבויות (hidden layers), ושכבת יציאה (output layer) כך שכל קודקוד, או בעצם.. נוירון מלאכותי, מתחבר לקודקודים של השכבה הבאה וחולק איתם קשת ממושקלת.

עבור כל הקלטים שנוירון מקבל, הוא עושה חישוב בעזרת פונקציית אקטיבציה ומחליט אם להעביר את המידע הלאה או לא.

רשתות אלו הן כלי חזק מאוד לפתרון בעיות נפוצות ומציאת תבניות וקשרים בדומה לאיך שמוח האדם עובד.

קיימים 3 מרכיבים עיקריים, ברשת העצבית של בני אדם, שעליהם מבוססת רשת הנוירונים המלאכותית, והם:

1. דנדריטים Dendrites - ערוצי הכניסה של נוירון מסוים מנוירונים שכנים, מידע חשמלי מתקבל ממספר נוירונים מקושרים בו זמנית.
2. גוף התא Soma - כאשר הזרם שמתקבל מהדנדריטים חוצה ערך סף מסוים, תא העצב "יורה" אות חשמלי, ואז מאתחל את עצמו בציפייה לאירוע חציית סף נוסף.
3. אקסונים Axons - דרכם תא העצב מעביר מידע חשמלי לנוירונים הבאים.

נוירון מלאכותי מדמה את אופן פעולת הנוירון האנושי בכך שהוא מקבל קלט יחיד שמקורו בנוירון אחד או מספר נוירונים המחוברים אליו מכיוון אחד, מפעיל עליו פונקצייה כלשהי(בהקבלה לחציית ערך הסף), ומעביר ערכים הלאה לנוירונים המחוברים אליו בכיוון השני.

באמצעות מודל זה ניתן ליצור משטחי הפרדה לא ליניאריים מורכבים ביותר.

בהמשך הפרוייקט, כאשר נרצה להשוות רשת זו לרשת קונבולוציה, נעשה שימוש ברשת נוירונים fully connected שבה כל נוירון בשכבה מסויימת מחובר לכל הנוירונים של השכבה הבאה.

### 3.3.2 רשתות נוירונים – שיטת עבודה

Forward and Back Propagation

האופן שבו רשת הנוירונים לומדת הוא על ידי העברת מידע רב ברשת ועדכון המשקלות בהתאם לפונקציית המחיר.

Diagram

Description automatically generatedForward Propagation - המידע עובר משמאל לימין, כאשר בשכבת הכניסה המידע מועבר כמו שהוא לשכבה הבאה (שכבה חבויה), בשכבה זו כל נוירון שמקבל מידע, מפעיל את פונקציית האקטיבציה שמוגדרת לו (לרוב מגדירים פונקציית אקטיבציה זהה לכל השכבה).  
הנוירון יעביר את המידע שהתקבל מפונקציית האקטיבציה הלאה לנוירונים הבאים.  
בצורה זו המידע ממשיך לעבור בין השכבות החבויות ולבסוף מגיע לשכבת היציאה בשביל קבלת ההחלטה באמצעות פונקציית ההיפותזה.

הדגמה מתמטית לForward Propagation:

\*פונקציית ההיפותזה זו היא פונקציית האקטיבציה של שכבת המוצא

Activation function – ישנן מספר פונקציות אקטיבציה שונות.

תפקידן העיקרי הוא לקחת את סכום ההכפלות של הערכים הנכנסים והמשקולות, לבצע עליו חישוב מתמטי כלשהו ולהעביר את התוצאה הלאה.

פונקציות האקטיבציה שבהן נעשה שימוש בפרויקט הן: ReLU ו- Softmax.

ReLU activation

Softmax activation

Diagram, schematic

Description automatically generatedText

Description automatically generated

Backpropagation – בתהליך זה נעדכן את המשקלות על ידי שימושי בעיקרון ה – Gradient descent למזעור פונקציית המחיר 𝐽(𝛩) .

A picture containing icon

Description automatically generatedחישוב הנגזרות החלקיות מאפשר לדעת את כיוון הירידה התלולה ביותר של 𝐽(𝛩) .  
לכן, בכל שלב נעדכן את הפרמטרים בצעדים הדרגתיים, בעזרת מקדם הלמידה 𝛼 לכיוון שהתקבל, אל עבר מינימום מקומי של 𝐽(𝛩) .

ערך העדכון שמתווסף לכל פרמטר נקרא הגרדיאנט ומוגדר כך:

Text

Description automatically generated with low confidence

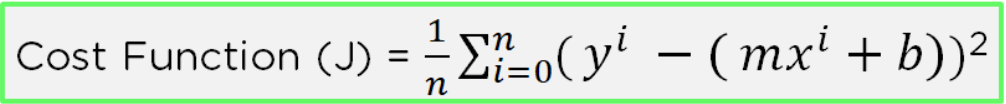
לכן, כלל העדכון לכל פרמטר בכל איטרציה יהיה מהצורה:

Text

Description automatically generated

Cost function – תפקידה של פונקציית המחיר הוא לחשב כמה רחוקה פונקציית ההיפותזה מהמציאות.

בשונה מבדיקת המודל, וקבלת תוצאות על הצלחתו, פונקציית המחיר עוזרת לנו גם להבין איך לשפר את המודל וזאת על ידי הטמעתה ב Gradient descent, עבור כל 𝛩 ברשת.

ברשתות נוירונים, לכל שכבה תהיה פונקציית מחיר משל עצמה, ולכל פונקציית מחיר יהיה ערך שגיאה מינימלי ייחודי(תלוי באיזו נקודה התחלנו).  
בסוף המטרה היא להגיע למינימום גלובלי בכל השכבות יחד ומינימום לוקאלי לכל שכבה לפי הנוסחה הבאה:

### 3.3.3 רשתות נוירונים – בדיקת מודל

* משטחים כל תמונה לשכבת כניסה של 12,250 נוירונים.
* הבדיקה תתבצע על רשת fully connected של 8 שכבות חבויות.
* שכבת היציאה תקבע לנו האם בקטע(תמונה) הייתה או לא הייתה ציפור.
* כדי לראות שהרשת באמת מתקשה ללמוד ושהיא לא נכשלה במקריות נריץ K-Fold Cross Validation\*.

\*K-Fold Cross Validation:

Model architecture:

**Input layer:**

Input – 12,250

**Hidden layers:**

Dense (2048) - 2048

Dense (1024) - 1024

Dense (512) - 512

Dense (512) - 512

Dense (256) - 256

Dense (128) - 128

Dense (64) - 64

Dense (32) – 32

**Output layer:**

Dense (2) - 2

חזרה על שלב הלמידה מספר פעמים עם חלוקה שונה   
 לקבוצות למידה ומבחן כדי לראות שארכיטקטורת הרשת  
 פועלת באופן יציב לאורך K בדיקות שונות על קבוצות  
 אימון שונות.  
 תהליך זה עוזר מאוד להילחם בבעיות כמו overfitting   
 בכך שניתן לראות שהרשת לא נוטה (biased) לכיוון אף  
 קבוצת אימון ספציפית.

### 3.3.4 רשתות נוירונים – תוצאות

Chart, line chart

Description automatically generated

Model accuracy

Chart, line chart

Description automatically generatedChart, treemap chart

Description automatically generated

ההסתכלות המרכזית תהיה על הConfusion Matrix ובפרט על אחוזי ה False positive ו- True positive:

* Confusion Matrix – טבלה ריבועית שבה כל עמודה ​​מייצגת את המופעים במחלקה חזויה ואילו כל שורה מייצגת את המופעים במחלקה בפועל
* False Positive Rate – אחוז הפעמים שהמערכת טעתה וניבאה שיש ציפור לעומת סך הפעמים שלא הייתה ציפור
* True Positive Rate – אחוז הפעמים שהמערכת ביצעה חיזוי נכון וניבאה שיש ציפור לעומת סך הפעמים שהייתה ציפור

Model loss

K-fold Cross Validation results:

Score per fold:

* Fold 1 - False Positive: 68.55 - Accuracy: 31.45%
* Fold 2 - False Positive: 67.35 - Accuracy: 32.65%
* Fold 3 - False Positive: 70.18 - Accuracy: 65.94%
* Fold 4 - False Positive: 18.18 - Accuracy: 71.22%
* Fold 5 - False Positive: 68.31 - Accuracy: 31.69%
* Fold 6 - False Positive: 68.77 - Accuracy: 31.23%
* Fold 7 - False Positive: 68.89 - Accuracy: 31.11%
* Fold 8 - False Positive: 40.03 - Accuracy: 74.78%
* Fold 9 - False Positive: 69.35 - Accuracy: 30.65%
* Fold 10 - False Positive: 67.28 - Accuracy: 32.72%

Average scores for all folds:

* Accuracy: 43.3439 (+- 17.9942)
* False Positive: 60.689 (+- 17.4456)

## 3.4 רשתות קונבולוציה – Convolutional neural network

### 3.4.1 רשתות קונבולוציה – הקדמה

בלמידה עמוקה, ישנן הרבה מחלקות של רשתות נוירונים לבעיות שונות עם סוגי מידע שונים למשל Recurrent neural network משמשת בעיקר לעיבוד וזיהוי שפות בעוד ש -

Convolutional neural network(רשת קונבולוציה), או CNN, משמשת בעיקר לראייה ממוחשבת וזיהוי תבניות בתמונות.

מטרתינו העיקרית היא לזהות בספקטרוגרמות(תמונות) צורות שמייצגות ציוצי ציפורים.

רשת קונבולוציה חוסכת לנו המון זמן בהוצאת מאפיינים, וניתוח התמונות באופן ידני, לפני הלמידה, בשונה ממה שנדרש לעשות ברשתות אחרות, והיא גם תעשה זאת, כנראה, יותר טוב ממה שאנחנו נעשה באופן ידני.

ננסה להראות שרשת זו טובה יותר לבעיה שלנו מרשת נוירונים fully connected.

רשת קונבולוציה מורכבת בעיקר משכבות שמבצעות קונבולוציות על התמונות, לומדות את המאפיינים ומעבירות את המידע הלאה, אך בנוסף, ברשת זו קיימות גם שכבות נוירונים fully connected כמו ברשת שבדקנו קודם לכן ומכאן גם שזמני הריצה ארוכים יותר וכמות הזיכרון גדולה יותר.

לכל שכבה נקבע מספר המסננים כך שכל מסנן אחראי להוצאת מאפיינים אחרים מהמידע.

בנוסף קיימות ברשת גם שכבות pooling שאחראיות למצות את המידע שסונן על ידי שכבות הקונבולוציה.

### 3.4.2 רשתות קונבולוציה – אופן פעולה

רשתות קונבולוציה נבדלות מרשתות נוירונים אחרות על ידי טיב ביצוען בעבודה עם תמונות ואותות קוליים.

יש להן 3 סוגים עיקריים של שכבות:

* שכבת קונבולוציה
* שכבת pooling
* שכבת fully connected

שכבת הקונבולוציה היא השכבה הראשונה ברשת.

בעוד שלשכבות קונבולוציה יכולות להתלוות שכבות קונבולוציה וpooling נוספות, שכבת ה- fully connected היא השכבה האחרונה.

עם כל שכבה, רמת הסיבוך של הרשת עולה ובכך מזהה יותר ויותר מאפיינים בתמונה.

השכבות הרדודות מתמקדות במאפיינים פשוטים, כמו צבע וקצוות, וככל שמתקדמים בעומק הרשת, מגלים מאפיינים עמוקים יותר עד שלבסוף מזהים את האובייקט הרצוי.

שכבת קונבולוציה

בשכבה זו מתבצעים רוב החישובים, ובעזרת קלט ופילטרים, היא מייצרת מפות מאפיינים ומעבירה אותן הלאה.  
בכל שכבה יכולים להיות מספר פילטרים כך שכל פילטר נע לאורך התמונה ויבדוק האם המאפיין שהוא מחפש קיים.  
הפילטר הוא מערך דו מימדי של משקולות אשר מייצגות חלק מהתמונה, בדרך כלל בגודל של 3x3.

משקלות הפילטר הן חלק מהפרמטרים שמתעדכנים לאורך אימון הרשת.  
בכל סוף פעולת קונבולוציה של שכבה, מופעלת פונקציית אקטיבציה על מפת המאפיינים.  
כמות הפילטרים נקבעת באופן ידני לכל שכבה בנפרד לפי סוג המידע, סוג הבעיה, ותוצאות הניסויים.

שכבת pooling

שכבות אלה נועדו כדי להקטין את כמות הפרמטרים בקלט, או במקרה שלנו, מקטינות את התמונה.  
בדומה לשכבות הקונבולוציה, גם בשכבות אלה יש שימוש בפילטר אך ההבדל הוא שלפילטר זה אין משקולות. במקום, פועלת פונקציית איסוף(aggregation), על הערכים שהתקבלו משכבות הקונבולוציה(מפות מאפיינים), ובכך מעבירות הלאה מטריצות חדשות קטנות יותר. בין היתר, ישנם 2 סוגים עיקריים של שכבות pooling:

הוצאת מקסימום – Max pooling

הוצאת ממוצע – Average pooling

שכבת fully connected

מעבר למה שכבר הוסבר על רשתות FC, ברשת קונבולוציה, תפקידן של שכבות אלו הוא לקיחת החלטות והסקת מסקנות על המידע שהועבר לה משכבות הקונבולוציה והpooling.

שכבות אלו יהיו בסוף הרשת כאשר השכבה האחרונה תשתמש בפונקציית אקטיבציה מתאימה לסוג הבעיה, למשל, עבור סיווג לא בינארי, נשתמש לרוב בsoftmax להוצאת הסתברויות הצלחה של כל אופציה.

### 3.4.2 רשתות קונבולוציה – ארכיטקטורת המודל

מטרתינו היא למצוא ארכיטקטורת מודל שתהיה טובה לבעיה שלנו.

מתחילים עם בדיקה של מספר מודלים על כמות קטנה יותר של קבצים כדי לקבל תחושה חיובית לגבי מודל מסוים.

התהליך הראשוני הזה הוא קצת בעייתי מכיוון שהמודל התעצב בהתאם לקבצים הספציפיים אותם בחרנו לבדיקה.

כדי להתגבר על בעיה זו מריצים K-fold cross validation על כל הקבצים למשך מספר קטן יותר של חזרות. עושים זאת כדי לקבל ביטחון כלשהו לגבי המודל ולאחר מכן אפשר לאמן את המודל על כל המידע בידיעה שהמודל כבר לא biased.

תהליך זה עלול להתבצע מספר פעמים עד לקבלת רשת מספקת.

בנוסף, לאורך הריצות מתגלות בעיות נוספות כמו overfitting ו- slow learning ולשם כך קיימות שכבות בKeras שעוזרות לטפל בבעיות אלה.

ביניהן אנו עושים שימוש בשכבות הבאות:

Dropout, SpatialDropout2D, GaussianNoise, BatchNormalization.

כאשר שכבות אלה, מלבד שכבת BatchNormalization, הן שכבות רגולריזציה אשר נועדו לטפל בבעיית ה Overfitting.

Overfitting

מצב שבו המערכת לומדת יותר מדי "טוב" את קבוצת האימון, כלומר, מצליחה לסווג טוב את קבוצת האימון אך פחות דוגמאות שעוד לא ראתה.  
מצב זה יכול לקרות ממספר סיבות וביניהן: כמות מידע לא מספקת, עומק הרשת גדול מדי, קבוצת האימון קטנה מדי ביחס לקבוצת המבחן, חלוקה לא טובה לקבוצות אימון ומבחן וכו'.

A picture containing text

Description automatically generatedבסוף כל ריצה של הרשת, פונקציית האופטימיזציה “Adam” מעדכנת את הפרמטרים כדי למזער את פונקציית המחיר “Cross Entropy” לפי הנוסחה הבאה:

לאחר כל התהליך, המודל המוצלח ביותר שמצאנו הוא מודל של שכבות קונבולוציה כפולות, עם ריפוד, הבנוי כך:

Model architecture:

Input - 175x70x1

Convolution (3x3) - 175x70x32

Convolution (3x3) - 175x70x32

Pool (3x3) - 58x23x32

Convolution (3x3) - 58x23x64

Convolution (3x3) - 58x23x64

Pool (2x2) - 29x11x64

Convolution (3x3) - 29x11x128

Convolution (3x3) - 29x11x128

Pool (2x2) - 14x5x128

Convolution (3x3) - 14x5x256

Convolution (3x3) - 14x5x256

Pool (2x1) - 7x5x256

Dense (256) - 256

Dense (32) - 32

Dense (2) – 2

Activation functions:

Hidden layers – ‘relu’

Output layer – ‘softmax’

בעל פונקציות האקטיבציה הבאות:

ReLU activation

Softmax activation

Diagram, schematic

Description automatically generatedText

Description automatically generated

התמודדות עם overfitting:

* שימוש בשכבות gaussian noise עם שונות של 0.1.
* הקטנת עומק הרשת.
* שימוש בשכבות dropout המשמיטות, בהסתברות מסוימת, חלק מהמידע.
* הגדלת המידע בעזרת דוגמאות חדשות / אוגמנטציות(נעשה לפני הלמידה).
* הוספת משקולת רגולריזציה לשכבה עם מספר הפרמטרים הגדול ביותר במודל.

התמודדות עם slow learning:

* שימוש בשכבות batch normalization אשר מנרמלות את הערכים במפות המאפיינים.

### 3.4.3 רשתות קונבולוציה – תוצאות

תוצאות הרצת K-fold cross validation:

K-fold Cross Validation results:

Score per fold:

* Fold 1 - False Positive: 15.83 - Accuracy: 91.36%
* Fold 2 - False Positive: 15.25 - Accuracy: 91.92%
* Fold 3 - False Positive: 15.84 - Accuracy: 91.53%
* Fold 4 - False Positive: 16.27 - Accuracy: 91.13%
* Fold 5 - False Positive: 14.75 - Accuracy: 91.09%
* Fold 6 - False Positive: 17.11 - Accuracy: 91.84%
* Fold 7 - False Positive: 17.27 - Accuracy: 91.39%
* Fold 8 - False Positive: 17.42 - Accuracy: 90.35%
* Fold 9 - False Positive: 13.98 - Accuracy: 90.81%
* Fold 10 - False Positive: 19.56 - Accuracy: 90.84%

Average scores for all folds:

* Accuracy: 91.226 (+- 0.459)
* False Positive: 16.328 (+= 1.59)

תוצאות האימון על המודל הסופי:

Chart, treemap chart

Description automatically generated

מבט על ה confusion matrix

מהתבוננות על המטריצה, ניתן לקבוע כי המערכת מצליחה לזהות ציפורים בדיוק של 88% (TPR).

לפי אחוז ה false positive, על כל 11 קטעי קול שבהם המערכת חזתה ציפור, היא טעתה ב 1 (FP)

Chart

Description automatically generated

Chart, line chart, histogram

Description automatically generated

מבט על הגרפים

ניתן לראות שהלמידה התבצעה באופן יציב, ולקראת חזרה מספר 20, החל להיווצר מצב של overfitting.

# פרק 4: מסקנות וסיכום

מסקנות

* רשת הנוירונים הרגילה לא הצליחה להתמודד עם הבעיה בשונה מרשת הקונבולוציה שטובה יותר בפתרון בעיות הקשורות לראייה ממוחשבת וזיהוי תבניות.
* השימוש בסולם מל וביצוע אוגמנטציות לתמונות שיפרו משמעותית את תוצאות הרשת.
* ככל הנראה היינו מגיעים לתוצאות טובות יותר אם היו לנו יותר דוגמאות אימון וחומרה טובה יותר (בעיקר כרטיס מסך).

המשך מחקר

* על מנת שנוכל להגיע לתוצאות טובות יותר נדרש להמשיך לתייג עוד קבצים כדי להגדיל את כמות דוגמאות האימון ולהקטין את תופעת הoverfitting.
* ניתן לקחת את תוצאות הרשת בתור דוגמאות אימון לרשת אחרת שתלמד לסווג גם את סוגי הציפורים.

סיכום

הצלחנו לאמן מערכת שתדע לקבוע, ברוב המקרים, הימצאות ציפור בקטע קול וזאת על ידי שימוש בתמונת הספקטרוגרמה של הקטע.

# ביבליוגרפיה

**DOMAIN TUNING METHODS FOR BIRD AUDIO DETECTION**

Technical Report by

*Sidrah Liaqat, Narjes Bozorg, Neenu Jose, Patrick Conrey, Antony Tamasi and Michael T. Johnson*

**from DCASE2018\_Kentucky.pdf**

**ACOUSTIC BIRD DETECTION WITH DEEP CONVOLUTIONAL NEURAL NETWORKS**

Technical Report by

*Mario Lasseck*

**from DCASE2018\_Lasseck\_76.pdf**

**https://www.simplilearn.com/tutorials/machine-learning-tutorial**

**https://www.machinecurve.com/**

**https://calvinfeng.gitbook.io/machine-learning-notebook/**

**https://machinelearningmastery.com/**