**שימוש בכלים של למידה חישובית לזיהוי של מיני   
ציפורים על-פי קולן**

מגישות:

גלי לבל – 203843743

שיר בר מעוז – 205848914

מנחה מלווה:

פרופ' יזהר לבנר

**תוכן עניינים**

מבוא ................................................................................................................................................ 3

CNN ................................................................................................................................................ 4

MNIST Database ........................................................................................................................ 6

FCN ................................................................................................................................................. 8

RNN ................................................................................................................................................ 9

מהלך העבודה שלנו .................................................................................................................... 10

מודל 1 – fully connected ....................................................................................................... 11

מודל 1 – FCN .............................................................................................................................. 14

סיכום ............................................................................................................................................. 18

ביבליוגרפיה .................................................................................................................................. 19

**מבוא**

מינים רבים של עופות משמשים בתפקידים חשובים לשמירת האיזון הטבעי וכמיני מודל להבנת תהליכים טבעיים וכביואינדיקטורים. למרות חשיבותם, רבים נמצאים בסכנת הכחדה. ניטור יעיל של שינויים באוכלוסיות נדרש על מנת לנהל בתי גידול ולשמרם. עם התקדמות הטכנולוגיה ושכלול שיטות ההקלטה והעיבוד, נעשה שימוש נרחב בהקלטות ועיבודן למטרות מחקר ושימור. אחד האתגרים החשובים כיום הוא ייצור תהליך ממוחשב לצורך זיהוי אוטומטי של מיני ציפורים מתוך הקלטות אקוסטיות של שירתן. בשנים האחרונות שיטות של "למידת מכונה", ענף מרכזי מתחום הבינה המלאכותית, משמשות למטרות סיווג נתונים רבים בצורה יעילה.

בפרויקט זה אנו מציעים אלגוריתם אוטומטי לסיווג מיני ציפורים לפי קולותיהן.

**למה בחרנו בפרויקט?**

כאשר ישבנו לראשונה לבחור פרויקט שבו נרצה לעסוק במהלך השנה, היה לנו ברור שאנחנו רוצות לבחור פרויקט שקשור ל-machine learning.

למידת מכונה הוא תחום העוסק במגוון משימות חישוביות בהן התכנות הקלאסי לא ישים. המאפיין המרכזי של למידת מכונה הוא בפיתוח אלגוריתמים שאינם מבוססים על סט חוקים מוגדר מראש, אלא לומדים מתוך מצבור דוגמאות. המטרה המרכזית של למידת מכונה היא טיפול ממוחשב בנתונים, על ידי מידול, חיזוי או גילוי עובדות מן העולם האמיתי, עבור בעיות שלא ניתן לכתוב להן תכנת מחשב "קלאסית".

בעולם הטכנולוגי שבו אנחנו חיים, כמעט לכל שאלה נמצאת תשובה בתוך הנייד שלנו. פעם כשהיינו הולכים לטיול, היינו מצטיידים בספר וכאשר היינו רואים חיה או פרח, מיד היינו בודקים בספר מי היא ומחפשים עליה פרטים. היום, בלחיצת כפתור אחת כבר אפשר לזהות את מיד את הפרח, את הציפור וכל מידע נוסף בתוך האנציקלופדיה האינסופית של הבינה המלאכותית.

מתוך סקרנות גדולה לדעת ולהבין כיצד העולם הזה עובד, בחרנו בפרויקט שעושה בלמידה עמוקה, עיבוד אותות ולמידה חישובית, כדי לנסות ולבנות מודלים שניתן לאמן אותם על בסיס סט גדול של אימון (train), ולאחר מכן בזמן אמת המודל יידע לחזות בדיוק איזה סוג ציפור זו, לפי שירתה.

העבודה אמנם הייתה קשה ולא פשוטה, אך מאוד מלמדת, מעניינת ומאתגרת, ויצאנו ממנה עם סט גדול של כלים לעתיד שלנו.

**Deep Convolutional Neural Networks**

רשת עצבית מתפתחת (CNN-Convolutional Neural Network), היא סוג של רשת עצבית המשמשת בעיקר לניתוח תמונה מלאכותית. הסוגים העיקריים של רשתות עצביות המשמשות לעיבוד אותות רב ממדים, הן CNN - רשתות עצביות מתפתחות עמוקות. המונח עמוק מתייחס באופן כללי לרשתות בעלות מספר שכבות קונבולוציה, ולמידה עמוקה מתייחסת למתודולוגיות להכשרת מערכות אלה כדי ללמוד באופן אוטומטי את הפרמטרים התפקודיים שלהן באמצעות נתונים מייצגים.   
רשתות CNN נמצאות כיום בשימוש במגוון רחב של תחומי יישום, שכולן חולקות את המטרה המשותפת של **היכולת ללמוד אוטומטית תכונות ממאגרי נתונים** (בדרך כלל מסיביים) **ולהכליל את תגובותיהם** לנסיבות שלא נתקלו בהן בשלב הלמידה.

זיהוי תבניות על ידי מכונה כולל את ארבעת השלבים הבסיסיים הבאים:

1. Acquisition - מייצר את נתוני הקלט (למשל, במקרה שלנו הקלטות שמע).

2. Preprocessing - עיבוד מוקדם עוסק במשימות כגון הפחתת רעש ותיקונים.

3. Feature Extraction - עוסק בתכונות מחשוב הבסיסיות להבדיל בין סוגי תבניות.

4. Classification – תהליך המקציב תבנית קלט נתונה לאחת ממספר מחלקות המוגדרות מראש.

רשתות CNN מציעות גישה חלופית שמאפשרת למידה של תכונות על ידי שימוש במאגרי מידע גדולים של דוגמאות, הנקראים "training set", המייצגים תחום יישומים מעניין.

היא מבוססת על שכבות כינוס (convolution layers) הממומשת ע"י העברת filter ל-karnel על גבי הקלט, וביצוע כפל מטריצות. הפלט נקרא feature map.

רשת עצבית מתפתחת עושה שימוש בדרך כלל בשכבות: fully connected, convolution ו-pooling. היא משתמשת בפונקציית ההפעלה ReLU.

**Convolution Layer**

הקלט לשכבה הזו הוא טנזור. טנזור הוא פונקציה מולטי-ליניארית. אפשר להגדיר אותו כהעתקה מולטי-ליניארית (כלומר פונקציה ממרחב וקטורי למרחב וקטורי ששומרת על החיבור והכפל בסקלר) של וקטורים ופונקציונלים אל שדה המספרים הממשיים R.

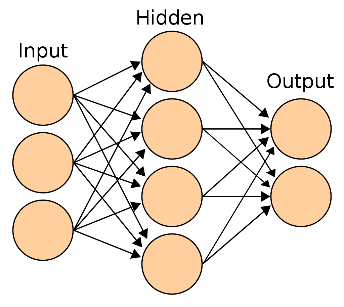
הטנזור הוא בגודל (מספר קלטים) X (מספר ערוצי קלט) X (גובה הקלט) X (רוחב הקלט).

לאחר שהקלט מסיים לעבור בשכבה מתקבל טנזור המייצג תמונה מופשטת יותר הנקראת feature map, בגודל (מספר קלטים) X (מספר ערוצי ה-feature map) X (גובה ה- feature map) X (רוחב ה-feature map).

שכבת הקונבולוציה מקבלת [טנזור](https://he.wikipedia.org/wiki/%D7%98%D7%A0%D7%96%D7%95%D7%A8" \o "טנזור) ומוציאה [טנזור](https://he.wikipedia.org/wiki/%D7%98%D7%A0%D7%96%D7%95%D7%A8" \o "טנזור) חדש שישמש כקלט עבור השכבת קונבולוציה הבאה, ממש כמו שכבות fully connected ב[רשת עצבית מלאכותית](https://he.wikipedia.org/wiki/%D7%A8%D7%A9%D7%AA_%D7%A2%D7%A6%D7%91%D7%99%D7%AA_%D7%9E%D7%9C%D7%90%D7%9B%D7%95%D7%AA%D7%99%D7%AA). אחד מיתרונות שכבות הכינוס הוא הוזלת עלות החישוב של תמונות וכתוצאה מכך אפשור העמקת הרשת.

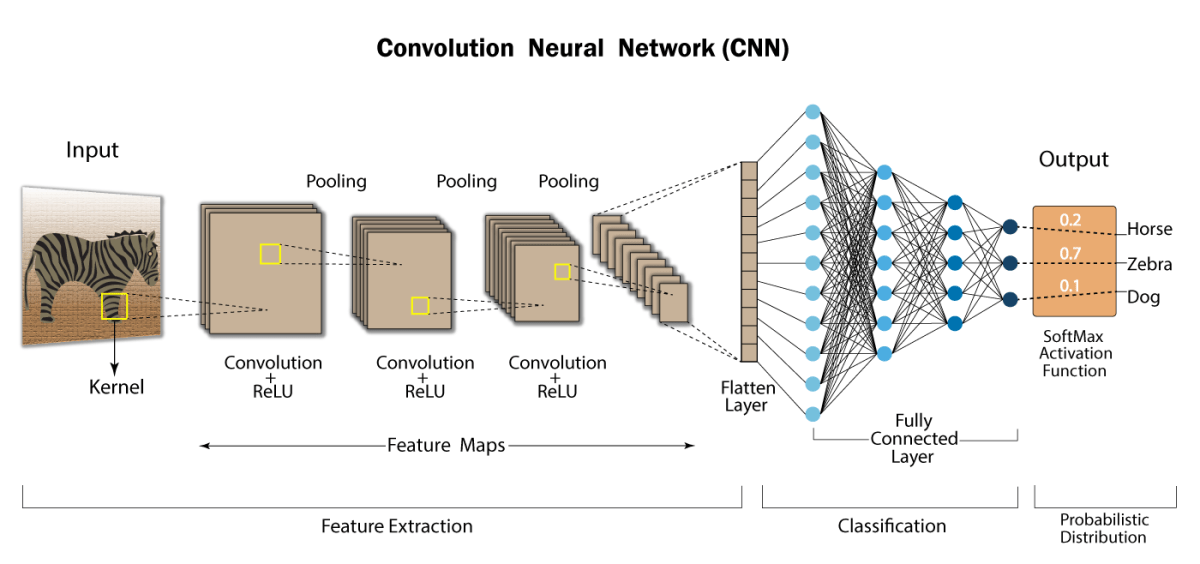
**Pooling Layer**

שכבה זאת נועדה על מנת לצמצם את גודל הטנזור. היא מקבלת [טנזור](https://he.wikipedia.org/wiki/%D7%98%D7%A0%D7%96%D7%95%D7%A8" \o "טנזור) בגודל מסוים בתור קלט, ומחזירה [טנזור](https://he.wikipedia.org/wiki/%D7%98%D7%A0%D7%96%D7%95%D7%A8" \o "טנזור) קטן יותר בתור פלט. יש כמה סוגי שכבות כאלה כמו: max pooling, average pooling ו-min pooling. סיבת השימוש בה היא בעיקר על מנת להוריד את "עלות" החישוב ולחסוך בזמן עיבוד.



**Fully Connected Layer**

מכילה מערך נוירונים. מקבלת כקלט מערך נוירונים ומחברת בין כל נוירון במערך זה לכל נוירון הנמצא בתוכה.

**תמונה להמחשה:**

**MNIST Database**

כדי לכתוב את המודל שלנו, שעתיד לסווג מיני ציפורים על פי שירתן, למדנו סוגים שונים של בעיות כדי שאלה ישמשו לנו בסיס ללמידה ולתרגול. זאת על מנת שנדע כיצד לפתח, להעריך ולהשתמש ברשתות עצביות ללמידה עמוקה מתפתחת ולסיווג תמונות, לחקור הרחבות למודל בסיסי לשיפור יכולת הלמידה והמודל, וללמוד כיצד לפתח מודל סופי בעצמינו לבעיה דומה.

אחד המאגרים הגדולים שנמצאים ברשת הוא MNIST. מאגר ה-MNIST הוא מאגר גדול של ספרות בכתב יד, המשמש בדרך כלל לאימון מערכות עיבוד תמונה שונות. המאגר נמצא בשימוש נרחב גם להכשרה ובדיקות בתחום של machine learning. הוא מאגר משנה של מערך הנתונים NIST, ומורכב מכתב ידם של עובדי הלשכה למפקד האוכלוסין האמריקאי ומכתב ידם של תלמידי תיכון אמריקאים. הספרות מנורמלות בגודל קבוע ובמרכז התמונה.

מאגר ה- MNIST מכיל 60,000 תמונות הדרכה ו -10,000 תמונות בדיקה.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

בקוד להלן אנחנו יכולים לראות תחילה טעינה של הנתונים מ-mnist לתוך מערכי אימון של תוויות ותמונות, ולתוך מערכי מבחן של תוויות ותמונות.

בלולאה אנחנו עוברים 35 פעמים על מנת להראות תמונה ותווית מתאימה.

בשורה 26 אנחנו מתחילים לבנות את המודל. המודל בו נשתמש יהיה Sequential, בו הפלטים או הקלטים מגיעים כרצף של נתונים. הם כוללים זרמי טקסט, שמע, וידיאו, נתונים מסדרות זמן וכו'. כיוון שבדוגמא זו מדובר על מערכי תמונות, ובגלל שבעבודה שלנו אנחנו עובדות עם קבצי אודיו- המודל אכן מתאים.

אנחנו מאמנים את הרשת בעזרת שכבת גודל בעלת 784 נוירונים (מטריצה של 28x28), בשכבה הנסתרת יהיו לנו 512 נוירונים ובשכבת הפלט יהיו לנו 10 נוירונים- אחד לכל ספרה.

משתמשים בפונקציית האקטיבציה ReLU, שהיא הפופולרית ביותר עבור רשתות עצביות עמוקות. זוהי פונקציית הפעלה ליניארית המוגדרת בין (0, ∞), שעבור כל קלט שהיא מקבלת – אם הוא חיובי היא פולטת אותו החוצה, אחרת תפלוט 0.

לאחר מכן נשתמש בפונקציית האקטיבציה softmax, שמוציאה מידת הסתברות. כל אחת מ-10 היציאות מוציאה מידת הסתברות בין 0 ל-1, כאשר סכום כל היציאות שווה ל-1.

בשורה 32 אנחנו מפעילים את המודל את compile, בעזרתה אנחנו מציינים את תצורת האימון, כלומר: האופטימיזציה (rmsprop), פונקציית האובדן (categorical\_crossentropy) והמימדים למעקב (accuracy).

לאחר מכן אנחנו עושים נרמול, כיוון שרשתות עצביות קוראות מידע סביב 0 ל-1, כלומר אנחנו צריכים לחלק ב-255 כדי שהתמונות יהיו שמורות ב- 8bits.

ואז בעזרת הפונקציה fit אנחנו מכשירים את המודל – הפונקציה מקבלת את מטריצות התמונות, מערך התוויות, ה-betch\_size ומחזורי הזמן (epochs). המודל מבצע חיתוך נתונים לקבוצות בגודל ה-betch\_size ועובר על הנתונים למספר הנתון של מחזורי הזמן.

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

ניתן לראות בסיום של כל מחזור זמן מה היה אחוז הדיוק (accuracy) וכיצד הוא משתפר ככל שהמודל מתאמן שוב ושוב על מערך התמונות, ולאחר 10 epoch אנחנו מגיעים להצלחה של 98 אחוז!

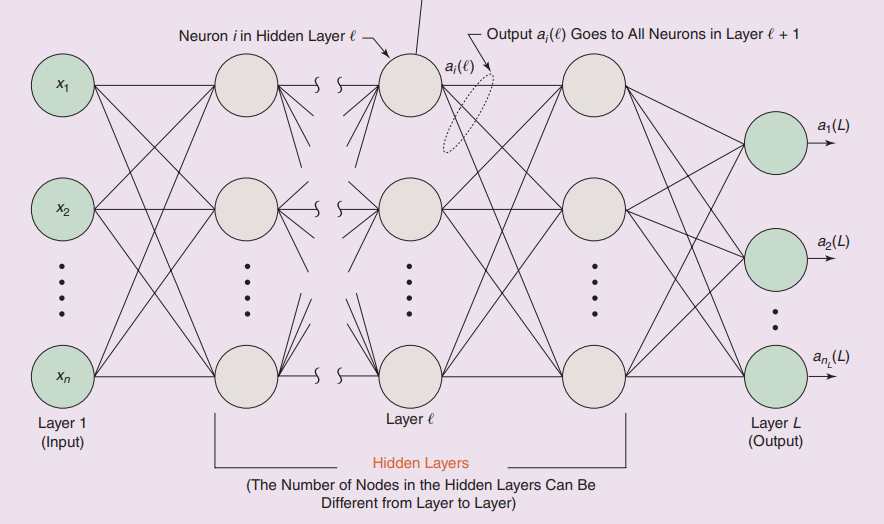
תמונה שמכילה חץ

התיאור נוצר באופן אוטומטיואכן כאשר מכניסים למודל המאומן מספרים, מתוך מאגר ה-test\_images ו-test\_lable, המודל מצליח לזהות את המספרים.

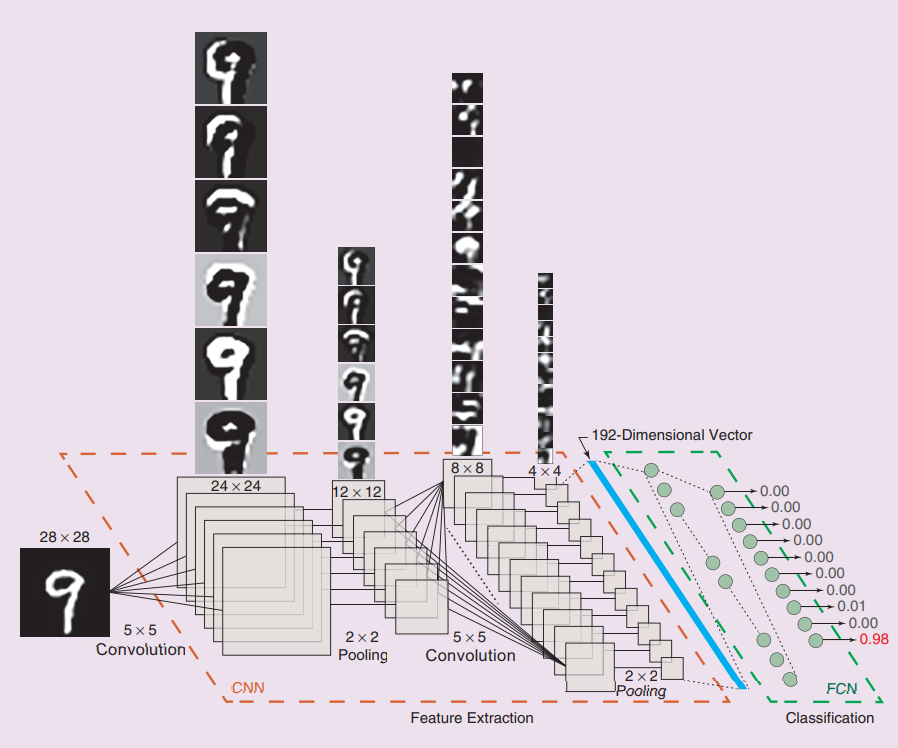
**FCN – Fully Convolutional Network**

Fully Convolutional מצביעה על כך שהרשת העצבית מורכבת משכבות קונבולוציה ללא fully connected layer (שכבות מחוברות לחלוטין) הנמצא בדרך כלל בסוף הרשת. רשת CNN עם fully connected layer הוא בדיוק כמו FCN עם שכבה אחת. ההבדל העיקרי הוא שב-FCN יש מסננים (filters) בכל מקום. אפילו בשכבה האחרונה שמוציאה את ה-output בנויה ממסננים.

רשת FCN מנסה ללמוד ייצוגים ולקבל החלטות על סמך הקלט המקומי בכל פעם. הוספת fully connected layer מאפשרת לרשת ללמוד משהו באמצעות מידע גלובלי במקום שבו הקלט נכשל.

**דוגמא לרשת fully connected (CNN):**

ניתן לראות בכל שכבה, שכל אחד מהנוירונים מחובר לכל הנוירונים שבשכבה הבאה.



**דוגמא לרשת FCN:**

פה אנו רואים רשת CNN שהוכשרה לחלץ תכונות המשמשות לאחר מכן את FCN לסיווג מספרים בכתב יד.

**RNN - Recurrent Neural Network**

רשת עצבית חוזרת (RNN) היא סוג של רשתות עצביות מלאכותיות (ANN) שבהן חיבורים בין צמתים יוצרים גרף מכוון לאורך רצף זמני. רשתות RNN יכולות להשתמש במצבן הפנימי (זיכרון) לעיבוד רצפי כניסות באורך משתנה. רשתות RNN שימושיות עבור משימות כמו זיהוי כתב יד מחובר או זיהוי דיבור.

קיימים מספר הבדלים בין רשתות CNN לרשתות RNN, העיקריים שבניהם:

* רשתות CNN משמשות בדרך כלל בפתרון בעיות הקשורות לנתונים מרחביים, כגון תמונות. רשתות RNN מתאימות יותר לניתוח נתונים זמניים ורציפים, כגון טקסט או סרטונים.
* רשתות CNN משתמשות במסננים ובשכבות pooling. לעומת זאת רשתות מסוג RNN עובדות בצורה כזו שהן מחזירות את ה-output בחזרה לרשת כ-input.
* ברשתות CNN גודל הקלט והפלט המתקבל הינם קבועים. כלומר, CNN מקבלת תמונות בגודל קבוע ומוציאה אותן לרמה המתאימה, יחד עם החיזוי המתאים. ברשתות RNN, גודל הקלט והפלט המתקבלים עשויים להשתנות.

**Long Short-Term Memory - LSTM**

רשתות LSTM מתאימות היטב לסיווג, עיבוד וחיזוי המבוססים על נתוני סדרות זמן, מכיוון שיכולים להיות פיגורים של משך זמן לא ידוע בין אירועים חשובים בסדרת זמן. היא סוג של ארכיטקטורת רשת עצבית חוזרת (RNN).

LSTM היא מערכת למידה עמוקה המונעת את בעיית "השיפוע הנעלם". כלומר, מונעת בעיית התפשטות של שגיאות. במקום זאת, טעויות יכולות לזרום לאחור דרך מספר בלתי מוגבל של שכבות וירטואליות שנפרשות במרחב. בעצם LSTM יכולה ללמוד משימות הדורשות זיכרונות מאירועים שהתרחשו אלפי ואפילו מיליוני שלבי זמן נפרדים קודם לכן. LSTM עובדת אפילו בהתחשב בעיכובים ארוכים בין אירועים משמעותיים, ויכולה להתמודד עם אותות המערבבים רכיבים בתדר נמוך וגבוה.

**מהלך העבודה על המודל שלנו**

**השלב הראשון**

תחילה למדנו חומר תיאורטי על סוגי רשתות עצביות: ANN, CNN, RNN.

הכרנו ספריות של python כמו librosa ו-sound device על מנת ללמוד את העבודה עם קבצים אודיו, איך לחלץ אותם, לבצע עליהן פעולות ועוד.

בנוסף, קיבלנו מהמנחה את הקבצים עמם החלנו לעבוד: 479 קבצי train ו-201 קבצי test. שתי הקבוצות מכילות 11 מיני ציפורים, ואורך הקלטות שנע בין שנייה ל-19 שניות.

**השלב השני**

בשל השני עברנו על כל ההקלטות על מנת לייצר עבור כל הקלטה mel spectrogram.

אותות אנלוגיים הם אותות שתוכן התדר שלהם משתנה לאורך זמן, כגון מוזיקה ודיבור. כדי לייצג את הספקטרום של אותות אלה, יש לבצע FFT (התמרת פורייה מהירה) בכמה מקטעי חלון של האות. ה-FFT מחושב על מקטעי חלון חופפים של האות, ואנו מקבלים את הספקטוגרמה.

בעת חישוב הספקטוגרמה, אנחנו הופכים את ציר ה-y לסקלה לוגריתמית, והאמפליטודה מומרת לדציבלים. הסיבה לכך היא כי בני אדם יכולים לתפוס רק טווח קטן ומרוכז של תדרים ואמפליטודות. ולכן, מבצעים פעולה מתמטית על התדרים כדי להפוך אותם להיות בסולם מל.

אבל לשם העבודה עם הספקטוגרמות, אנחנו רוצים סט אימונים עם הקלטות באורך שווה עבור כל הנתונים שלנו. לשם כך, תחילה בדקנו מה ה-median של כל ההקלטות.

אורך ה-median שנבחר היה 0.58 מילי-שניות, כיוון שרוב ההקלטות הן מאוד קצרות.

המשמעות של כך היא שעבור כל ההקלטות הארוכות יש לבצע חיתוך, ועבור כל ההקלטות הקצרות מהזמן שנבחר יש לבצע ריפוד באפסים.

בשלב הבא לאחר שעברנו על מערכי האימונים ומערכי המבחנים, ויצרנו לכל הקלטה ספקטוגרמה, שמרנו בקובץ dataFrame.pkl את המידע הדרוש לנו לשלבים הבאים: עבור כל הקלטה שמרנו את סיווג הציפור (ספרה בין 1-11), את המטריצה המייצגת את הספקטוגרמה עבור כל הקלטה בגודל (35 x 26), ואת המטריצה כווקטור של (1,910).

לאחר שהרכבנו את קבצי ה-dataFrame עבודה ה-test וה-train, יכולנו לבנות את המודלים על בסיס המידע ששמרנו ב-dataFrame במקום לעשות בכל פעם את הפעולות של קריאת הקבצים, יצירת ספקטוגרמות וכו' על מנת לאמן מודל.

**שלב שלישי – בניית מודלים**

**Model 1 - Fully Connected Network**

בשלב זה התחלנו בבניית הרשת fully connected. ברשת זו כל נוירון בשכבה אחת מחובר לכל הנוירונים בשכבה הבאה.

קוד המקור שלנו התבסס על המודל של mnist (שהוצג בעמוד 6).

הטבלה הבאה מציגה את מיני הציפורים אותן ניסינו לזהות ואת כמות הקלטים עבור מערך האימונים ועבור מערך הבחינה:

|  |  |  |  |
| --- | --- | --- | --- |
| **Bird** | **Test** | **Train** | **Label** |
| פשוש - Prinia gracilis | 38 | 89 | **1** |
| ירגזי - Parus major | 22 | 53 | **2** |
| דרור - Passer domesticus | 18 | 43 | **3** |
| שחרור - Turdus merula | 6 | 16 | **4** |
| ירקון - Chloris chloris | 14 | 32 | **5** |
| חוחית - Carduelis carduelis | 13 | 32 | **6** |
| צופית - Cinnyris osea | 26 | 61 | **7** |
| שלד לבן חזה - Halcyon smyrnensis | 36 | 84 | **8** |
| בולבול צהוב שת - Pycnonotus xanthopygos | 12 | 32 | **9** |
| קנית קטנה - Acrocephalus scirpaceus | 11 | 26 | **10** |
| קנית אירופאית - Acrocephalus arundinaceus | 5 | 11 | **11** |

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

בשלב הראשון של הקוד

אנחנו מבצעות חילוץ של הנתונים מתוך ה-dataFrame. אנחנו מגדירות את x\_train   
ו-x\_test בעזרת חילוץ המטריצה, ושומרות אותה כמערך מסוג float32.

בנוסף אנחנו מחלצות את   
ה-label שמייצג את אחת מתוך 11 סוגי הציפורים שהמודל מזהה.

את אותה פעולה בדיוק אנו עושות עבור נתוני ה-Test.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיבשלב הבא אנו מבצעות אימון של המודל.

השינויים שנעשו כאן, בשונה מה-MNIST המזהה תמונות, הוא חלוקה של המטריצות, המייצגות ספקטוגרמות,   
ב-(-80).

בעזרת fit אנחנו מאמנות את המודל עם נתוני ה-train, ולאחר מכן כדי לדעת מה אחוז הזיהוי הסופי בעזרת evaluate אנו בודקים מה אחוז החיזוי.

כדי להגיע לתוצאה הטובה ביותר, בדקנו מה יהיו ה-epochs וה-betch size.

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטיתמונה שמכילה טקסט, צילום מסך, מסמך

התיאור נוצר באופן אוטומטיepochs=50, betch size=64:

56%. לא כל כך מוצלח. אנחנו יודעת שסט האימונים שלנו הוא יחסית קטן, לכן כדאי להקטין את מספר ה-betch\_size.

epochs=50, betch size=32:

תמונה שמכילה טקסט, צילום מסך, מסמך

התיאור נוצר באופן אוטומטיתמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

באמת הצלחנו להשיג תוצאה של 68%. אבל 50 אימונים זה מעט ולכן ננסה להגדיל את המספר בשביל להביא לתוצאה טובה יותר.

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטיתמונה שמכילה שולחן

התיאור נוצר באופן אוטומטיepochs=150, betch size=32:

הגדלה משמעותית במספר ה-epoch אכן הביאה לתוצאה הרבה יותר טובה של 80%.

הגדלה נוספת של ה-epoch לא הביאה לעלייה באחוז הזיהוי.

אפשר לומר שאם היינו רוצות להשיג תוצאות טובות יותר היינו צריכות להכניס מספר גדול יותר של ספקטוגרמות כדי שהמודל יזהה טוב יותר בין מינים שונים של ציפורים.

**Model 2 - FCN – Fully Convolutional Network**

לאחר שביצענו את בניית מודל ה-fully connected, רצינו לראות מה יהיו התוצאות עבור מודל שמשתמשת ברשת קונבולוציה.

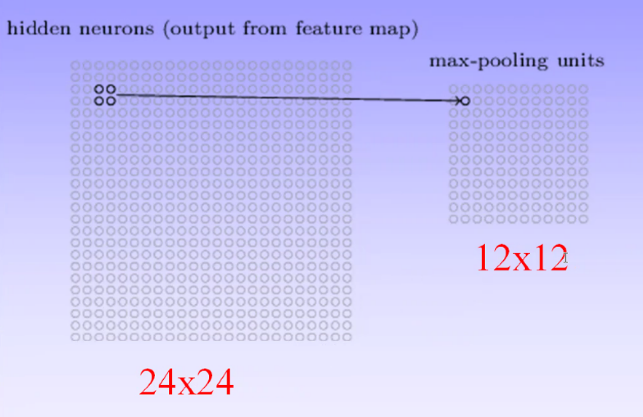
גם הפעם לקחנו קוד מקוד של mnist וביצענו עליו שינויים על מנת שיתאים למודל שלנו.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

תחילה אנו בונים את המודל, ע"י הגדרת הגודל לפי נתוני הקלט שלנו – כלומר מטריצה של (35x26), ושהרמה תהיה 1.   
לאחר מכן אנחנו מעבירים לרשת הקונבולוציה את השכבה הראשונה, תמונות המייצגות ספקטוגרמות. זוהי קונבולוציה דו מימדית, שמכילה 32 פילטרים, כל אחד מהם בגודל 3x3, ומעבירים לה את פונקציית האקטיבציה ReLU. כעת נשים את התוצאה ב-x ונעביר אותו לשכבה הבאה.

כאשר מדברים על 32 פילטרים, הכוונה היא שהרשת מוציאה 32 מפות תכונה של ערכי הכניסה. נסביר על זה בהמשך.



השכבה הבאה היא max pooling, בה נבצע הפחתת מימדים (=דחיסת נתונים). בשלב זה מספר הטרנספורמציות גדל פי 2, המשמעות היא אנחנו מקטינים את המטריצה פי 2 – כל ארבעה נוירונים אנו בוחרים את הנוירון בערך הערך המקסימלי והוא זה שיבחר לעבור לשכבה הבאה.

אנו חוזרים על השלב הזה 3 פעמים, כאשר בכל פעם מספר הפילטרים עולה, והמטריצה נעשית קטנה יותר.

לאחר מכן אנחנו מבצעים שיטוח לנתונים – הפקודה flatten עושה שיטוח למטריצה שמתקבלת ביציאה, כלומר הופכת אותה לוקטור של מספרים.

כעת נגדיר את ה-output. לוקחים את הוקטור x עליו ביצענו שיטוח, ומחברים אותו ל-11 היציאות שלנו, יציאה אחת עבור כל ציפור.

כעת נבצע חילוץ של המטריצות והתוותיות מתוך קובץ ה-dataFrame בדיוק כמו שביצענו במודל מספר 1.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיבשלב הזה נבצע אימון של המודל.

רשת הקונבולוציה, כשמה Conv2D מקבלת מטריצה דו מימדית, ולכן נעשה reshape על מנת להגדיר את הקלטים כ-479 מטריצות בגודל של (35x26) עבור ה-train, ו-201 מטריצות מטריצות בגודל של (35x26) עבור ה-test.

לבסוף נפעיל את המודל את compile, בעזרתה אנחנו מציינים את תצורת האימון, כלומר את האופטימיזציה (rmsprop), פונקציית האובדן (sparse\_categorical\_crossentropy) והמימדים למעקב (accuracy), כפי שהוגדר עבור ה-mnist בקובץ המקור.

גם הפעם כדי להגיע לתוצאה הטובה ביותר, בדקנו מה יהיו ה-epochs וה-betch size.

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

epochs=50, betch size=64:

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

83%. אמנם כבר עכשיו זה תוצאה יותר טובה מהמודל הקודם, אבל ננסה לשנות את הנתונים כדי להביא לתוצאה טובה יותר.

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטיepochs=100, betch size=64:

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

87%. נבצע ניסיון נוסף.

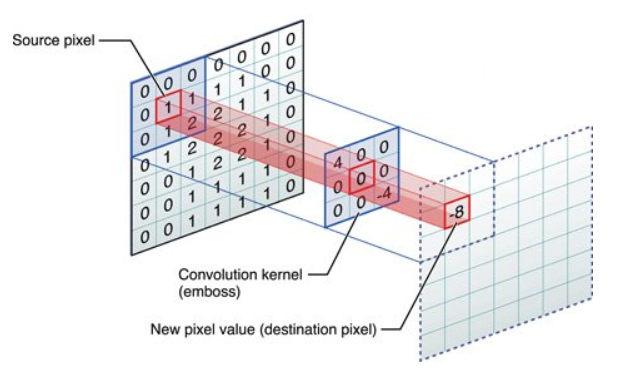
תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטיתמונה שמכילה שולחן

התיאור נוצר באופן אוטומטיepochs=150, betch size=64:

הגענו ל-89.6%, וניתן לומר שזאת התוצאה הטובה ביותר של חיזוי אליה הצלחנו להגיע.

**הסבר על ה-Summery ותכונות המפות:**

פעולת קונבולוציה בשני מימדים מבצעת את התהליך הבא:  
הפילטר המרכזי בתמונה בצד, שבגודל 3x3, נקרא kernel. פעולת הקונבולוציה מבצעת מכפלה פנימית בין ה-kernel לבין כל הפיקסלים בקלט אותו קיבלנו.

את התוצאה אנחנו שמים ב-feature map במקום המתאים. לאחר הפעולה הזו, המטריצה שתתקבל תהיה בגודל (n-2)x(m-2).

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטיבדוגמא בצד ניתן לראות את התרשים ה-output לאחר ביצוע כל שכבה. אנו רואים שבשכבה הראשונה קיבלנו מטריצות בגודל (35x26).

לאחר מכן ביצענו Conv2D אשר בה נוצרו 32 מפות, המטריצה שתתקבל תהיה בגודל (33x24), כמו שהסברנו לעיל. פעולת ה-max pooling, אשר מביאה להקטנת המימדים תביא את הפלט להיות בגודל (16,12,32).

כאשר נעבור לשכבה הבאה, אנחנו מגדילים את מספר הפילטרים, ומפעילים את הקלט שוב את Conv2D, ולכן יהיו לנו 64 מפות, כאשר מימדי המטריצה יהיו בגודל (14,10,64).

על הפלט נבצע max pooling ולכן הגודל יהיה (7, 5, 64). [נעביר את הפלט כקלט לשכבה נוספת עליה נבצע Conv2D, הפעם עם 128 מפות ולכן לאחר הפעולה הפלט יהיה בגודל (5,3,128).

על התוצאה האחרונה נבצע שיטוח, ונקבל וקטור בגודל (1,1920).

הסיבה שהגודל הוא 1920 כי אנחנו מכפילים את מימדי הפלט האחרון: 5 x 3 x 128.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

בנוסף, ביצענו בדיקה באופן ידני על מנת לבדוק האם כאשר אנחנו מכניסות ציפור ספציפית, המודל מזהה אותה כשורה.

אכן ניתן לראות, שאת השחרור, שהתווית שלו היא מספר 4, המודל לא זיהה וסיווג אותה לציפור הבולבול.

מבחינת נתוני המודל, בסט האימונים עבור השחרור קיימות רק 16 הקלטות, ובסט נתוני המבחן קיימות 6 הקלטות.

גם עבור הקנית האירופאית, שהתווית שלה היא מספר 11, המודל לא זיהה וסיווג אותה לקנית קטנה.

גם מבחינת נתוני האימון של הקנית האירופאית, בסט האימונים קיימות רק 11 הקלטות, ובסט נתוני המבחן קיימות רק 5 הקלטות.

אכן לא ציפינו לזהות את כל הציפורים כשורה, כיוון שהמודל נותן לנו רק אחוז זיהוי של 89%.

**סיכום**

כאשר בחרנו את הפרויקט, הגענו עם מעט מאוד ידע קודם על התחום אבל עם המון מוטיבציה ורצון ללמוד ולהבין איך מודלים של למידה עמוקה עובדים.

תחום הלמידה העמוקה, המשתמש בכלים של עיבוד אותות ולמידה חישובית, הוא אחד התחומים החשובים שמשתמשים בו כמעט בכל תחום טכנולוגי כיום. עניין אותנו לדעת, איך אפליקציה לזיהוי סוגי פרחים, למשל, עובדת. איך אפשר באמצעות צילום של פרח, לדעת ברגע האמת באיזה סוג מדובר. איך אפשר לבנות אפליקציה שבאמצעות האזנה למספר צלילים, מזהה את השיר המתנגן. ומכאן יצאנו למסע הזה, כדי ללמוד ולגלות דברים שיתנו לנו הצצה לתחום וכלים לעבודה בעתיד.

אחד השלבים הראשונים שעשינו היה למידה תאורטית – מהי רשת קונבולוציה, איך משתמשים בה ואיך ניתן לייצג אותה. למדנו איך להשתמש בספריות של פייתון וכמה הן חשובות, כיצד מייצגים צלילים באמצעות ספקטוגרמות ואיך ניתן להפוך אותן למטריצות ולעבוד עליהן בדיוק כמו שעובדים על תמונות.

הכרנו את ה-database של mnist ומודלים שונים לפתרון הבעיה, וניסינו להמיר את בעיית סיווג המספרים לפי תמונות לבעיית סיווג מין הציפור לפי זיהוי קולן.

הכרנו סוגים שונים של רשתות עצביות ועמדנו על ההבדלים בניהן. את הרשת העצבית RNN ואת המודל לזיהוי תנועות יפניות באמצעות רשת עצבית מסוג LSTM. לקראת סוף העבודה, הגענו גם למסקנה שיכול היה להיות מעניין לבדוק האם בניית רשת RNN תוכל להביא לתוצאות טובות יותר מאשר התוצאות שקיבלנו בעת הניסויים עם רשתות CNN.

אחד האתגרים שהיו לנו בדרך היה העבודה עם הספריות השונות של פייתון אותן לא הכרנו לפני תחילת העבודה. אתגר נוסף שהיה עבורנו היה שמירת הנתונים וחילוץ הנתונים לעבודה, אבל למדנו גם להתמודד עם בעיות כאלה ולהצליח לפתור אותן.

הפרויקט סוף פתח בפנינו דלת, ואנחנו מרגישות שיצאנו עם המון כלים להמשך הדרך.

אנחנו רוצות לומר תודה רבה לפרופסור יזהר לבנר, המנחה שלנו. שבסבלנות רבה והמון השקעה מזמנו לימד אותנו כל-כך הרבה דברים שלא יכולנו ללמוד באף מקום אחר.

עבורנו זאת הייתה זכות גדולה.

**ביבליוגרפיה**

הבדלים בין RNN ו-CNN:

<https://searchenterpriseai.techtarget.com/feature/CNN-vs-RNN-How-they-differ-and-where-they-overlap>

Article: A simple 2D CNN for MNIST digit recognition

<https://towardsdatascience.com/a-simple-2d-cnn-for-mnist-digit-recognition-a998dbc1e79a>

Article: How to Develop a CNN for MNIST Handwritten Digit Classification

<https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-from-scratch-for-mnist-handwritten-digit-classification/>

Sequence Classification Using Deep Learning

<https://www.mathworks.com/help/deeplearning/ug/classify-sequence-data-using-lstm-networks.html>

Article: Understanding the Mel Spectrogram

<https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53>

מידע על הציפורים:

פשוש - <https://he.wikipedia.org/wiki/%D7%A4%D7%A9%D7%95%D7%A9>

ירגזי - <https://he.wikipedia.org/wiki/%D7%99%D7%A8%D7%92%D7%96%D7%99_%D7%9E%D7%A6%D7%95%D7%99>

דרור - <https://he.wikipedia.org/wiki/%D7%93%D7%A8%D7%95%D7%A8_%D7%94%D7%91%D7%99%D7%AA>

שחרור - <https://he.wikipedia.org/wiki/%D7%A9%D7%97%D7%A8%D7%95%D7%A8>

ירקון - <https://he.wikipedia.org/wiki/%D7%99%D7%A8%D7%A7%D7%95%D7%9F_(%D7%A2%D7%95%D7%A3)>

חוחית - <https://he.wikipedia.org/wiki/%D7%97%D7%95%D7%97%D7%99%D7%AA>

צופית - <https://he.wikipedia.org/wiki/%D7%A6%D7%95%D7%A4%D7%99%D7%AA_%D7%91%D7%95%D7%94%D7%A7%D7%AA>

שלדג לבן חזה - <https://he.wikipedia.org/wiki/%D7%A9%D7%9C%D7%93%D7%92_%D7%9C%D7%91%D7%9F-%D7%97%D7%96%D7%94>

בולבול צהוב-שת - <https://he.wikipedia.org/wiki/%D7%91%D7%95%D7%9C%D7%91%D7%95%D7%9C_%D7%A6%D7%94%D7%95%D7%91-%D7%A9%D7%AA>

קנית קטנה - <https://he.wikipedia.org/wiki/%D7%A7%D7%A0%D7%99%D7%AA_%D7%A7%D7%98%D7%A0%D7%94>

קנית אירופאית - <https://he.wikipedia.org/wiki/%D7%A7%D7%A0%D7%99%D7%AA_%D7%90%D7%99%D7%A8%D7%95%D7%A4%D7%99%D7%AA>