

## מסמך מסכם - פרויקט גמר של הקורס רשתות נוירונים ולמידה עמוקה

מגישים: ניב אלכס (322822602), עדן אליהו (208081489), סיגל שחק (318955549)

נושא: ביצוע Multiclass and Multilabel Classification על תמונות של פוקימונים

### תקציר:

בפרויקט הגמר של הקורס רשתות נוירונים ולמידה עמוקה ביצענו שתי משימות מרכזיות העוסקות ב-Classification של תמונות פוקימונים.

### חלק ראשון – משימה של Multiclass classification על פוקימונים מדור 1

במשימה הראשונה עבדנו עם דאטהסט מאתר [Kaggle](#) הכולל כ-17,000 תמונות מתוגות של 150 פוקימונים **מדור 1**. המטרה המרכזית שלנו במשימה הראשונה הייתה לבצע משימה של Multiclass classification ובהינתן תמונה חדשה של פוקימון (מדור 1) שהמודל עוד לא ראה, לחזות לאיזה class הפוקימון הזה שייך (במקרה הזה יש 150 מחלקות שונות שמייצגות את מספר הפוקימונים השונים הקיימים בדור 1). מטרת העבודה הייתה לשחזר תוצאות בסיס (Baseline), לבצע תהליך חקירה ושיפור של מודלים כולל הטמעה של עקרונות מרכזיים שנלמדו לאורך הקורס כגון: Transfer Learning, התמודדות עם אתגרי Overfitting וחוסר איזון בין מחלקות, ולנתח את ביצועי המודל על קבוצת Test באמצעות מדדי דיוק Confusion Matrix, וניתוח אתגרים וחסימים.

תהליך העבודה כלל מספר שלבים: תחילה ביצענו בדיקות תקינות ואיכות של הנתונים כלומר זיהוי תמונות פגומות או בלתי קריאות ובדיקות שפיות קטנה להצגת דוגמאות של תמונות מתוך הדאטה, ולאחר מכן ניתוח התפלגות המחלקות כדי להבין את רמת חוסר האיזון (טווח של עשרות עד מאות תמונות למחלקה).

לאחר מכן, הנתונים חולקו לסט אימון, סט ולידציה וסט מבחן ביחס של 80/10/10 בצורה עקבית, ובנינו pipeline יעיל בפיתוח באמצעות tf.data (batch/prefetch) לשיפור מהירות האימון ויציבותו.

כחלק מההתמודדות עם חוסר איזון במחלקות הוספנו דוגמאות אימון חדשות באמצעות **Offline Augmentation** יצירת תמונות חדשות ושמירתן לסט האימון בלבד, עד ליעד אחיד יותר של תמונות למחלקה. יישמנו גם שיטות למלחמה ב-Overfitting כגון Dropout ושימוש ב-Callbacks כמו: EarlyStopping, ReduceLROnPlateau ו-ModelCheckpoint לשמירת המודל הטוב ביותר על בסיס סט הוולידציה.

בשלב יצירת המודלים נבנתה רשת Baseline CNN קטנה לצורך השוואה, ולאחר מכן הורצו מודלים מתקדמים המבוססים על עקרונות של למידת העברה (Transfer Learning) מארכיטקטורות מוכרות שהכרנו במהלך הקורס כגון EfficientNet ו-MobileNetV3 כאשר ביצענו fine-tuning לשכבות האחרונות במידת הצורך, יחד עם כיוונון Hyperparameters שונים ומגוונים כגון dropout, learning rate ולעיתים גם label smoothing.

כל הרצה תועדה בטבלת ניסויים הכוללת שם ריצה, זמן ריצה ותוצאות Val/Test לצורך ביצוע השוואה מקיפה בין המודלים השונים.

הערכת המודל בוצעה על קבוצת ה-Test בלבד בסיום התהליך, והוצגו גם גרפים של Accuracy/Loss לאורך האימון, מטריצת בלבול (Confusion Matrix) ניתוח זוגות בלבול נפוצים, וכן דוגמאות של הצלחות וכישלונות של המודל הסופי. המודל המיטבי היא מודל של EfficientNet שבוצע עליו fine tune והגיע לדיוק של Test של 0.864 כאשר מרבית הטעויות הופיעו בין פוקימונים בעלי דמיון חזותי גבוה (ממחלקות קרובות), בעוד שבמחלקות מובחנות המודל הציג זיהוי יציב ובטוח.

## חלק שני – ביצוע משימה של Multilabel classification על פוקימונים מדורות 1-7

במשימה השנייה הלכנו צעד אחד קדימה וניסינו לבצע משימה של multi-label classification לתמונות של פוקימונים (מהדור הראשון ועד הדור השביעי). לקחנו תמונות של 809 פוקימונים מאתר [Kaggle](#) ועל בסיס בנינו מודלים שאמורים לחזות את הסוג הראשי והסוג המשני של פוקימון חדש. סך הכל קיימים 18 סוגי פוקימונים אפשריים: רגיל, אש, מים, חשמל, עשב, קרח, לחימה, רעל, אדמה, מעופף, פסיכי, חרק, סלע, רוח, דרקון, אופל, פלדה ופיה. פוקימון יחיד יכול להיות שייך לסוג אחד ראשי בלבד (למשל הפוקימון פיקאצ'ו הוא פוקימון מסוג חשמל בלבד) או לסוג ראשי ולסוג משני (למשל הפוקימון בלבאזור שייך לסוג הראשי עשב ולסוג המשני רעל).

מקור הנתונים שלנו למשימה השנייה היה מאגר נתונים של תמונות פוקימונים מאתר Kaggle הכולל **809 תמונות** של פוקימונים מדור 1 ועד דור 7, כאשר כל תמונה של פוקימון עברה עיבוד מקדים שלנו ונקבעה לגודל אחיד של 120 פיקסלים על 120 פיקסלים בפורמט RGB לצורך אימון המודלים.

בנוסף לאוסף של 809 התמונות, מאגר הנתונים שלנו כולל גם קובץ CSV המאגד מידע טבלאי ו-labels של סוגים עבור כל פוקימון. קובץ ה-CSV מכיל 809 רשומות של פוקימונים ומאפשר לנו לבצע התאמה עקבית בין כל תמונה לבין התוויות (תוויות כלומר הסוג הראשי של הפוקימון והסוג המשני של הפוקימון אם קיים) המתאימות לה.

הבעיה שאנחנו מנסים לפתור במשימה זאת מוגדרת כבעיית Multi-Label Classification משום שלכל פוקימון קיימת אפשרות להשתייך לסוג אחד או שני סוגים לכל היותר: סוג ראשי (Primary type) ולעיתים גם לסוג משני (Secondary type).

הגדרה זו מאתגרת יותר מבעיית Multi-Class רגילה, שכן המודל שלנו נדרש להפיק בזמנית מספר תוויות נכונות ולהתמודד עם תלות אפשרית בין labels שונים, ולספק הסתברויות במקום לבחור מחלקה יחידה מנצחת.

בעוד שעבודות קודמות בתחום עסקו בחיזוי סוגי פוקימונים מתוך תמונות, רבות מהן התמקדו בעיקר בחיזוי **הסוג הראשי בלבד**. במשימה זאת לקחנו את לקחנו את האתגר צעד קדימה וניסינו לחזות גם סוג ראשי וגם סוג משני, ובכך הגדלנו את מורכבות הבעיה.

תהליך העבודה שלנו כלל מספר שלבים. בשלב הראשון ביצענו עיבוד מקדים לתמונות של הפוקימונים (קריאה מהדאטה-סט, שינוי גודל התמונה ונרמול). לאחר מכן התאמנו לכל תמונה של פוקימון את ה-label המתאים לה מקובץ ה-csv כאשר ה-label הוא רשימת הסוגים (ראשי ומשני) של הפוקימון הספציפי המדובר. בשלב הבא, עבור כל label ביצענו קידוד לייצוג Multi-Hot כדי להכין את הקלט למודל.

בשלבים הבאים ביצענו ניתוח ראשוני של התפלגות סוגי הפוקימונים, ובניית מודלים שונים הכוללים רשתות CNN קלאסיות ומודלים של למידת העברה (Transfer learning). אתגר מרכזי שזיהנו במשימה זו הוא חוסר איזון בנתונים: סוגים מסוימים של פוקימונים מופיעים בתדירות גבוהה בהרבה מאחרים, דבר שעלול להטות את המודל לטובת מחלקות נפוצות.

התוצאות הסופיות הראו כי שימוש בלמידת העברה (Transfer Learning) באמצעות MobileNet שיפר את ביצועי המודל ביחס למודל הבסיסי. למרות שהביצועים נותרו מוגבלים, מדד ה-Top-K Categorical Accuracy הגיע ל-35% ערך הגבוה יותר בהשוואה לבייסליין. ממצא זה מצביע על כך שהייצוגים החזותיים שנלמדו מראש במודל המאומן תורמים ליכולת טובה יותר לדרג את הקטגוריות הנכונות של הפוקימונים בין התחזיות המובילות, גם כאשר המשימה מורכבת (דאטה סט קטן יחסית) והנתונים מאתגרים יחסית (תמונה יחידה של כל פוקימון, בניגוד לחלק הראשון של הפרויקט).

## **חלק א' של הפרויקט**

### **רקע לבעיה הראשונה – בעיית Multiclass Classification :**

סיווג תמונות של פוקימונים מהווה אתגר מעניין בתחום הלמידה העמוקה, בעיקר בשל השילוב בין **דמיון חזותי בין מחלקות שונות** לבין **שונות גבוהה בתוך אותה מחלקה**. מצד אחד, פוקימונים רבים חולקים מאפיינים דומים (צבעים, מבני גוף, עיניים/אוזניים, קווי מתאר ואפילו סגנון ציור), ובמיוחד כאשר מדובר בהתפתחויות של פוקימון או פוקימונים מאותה משפחה חזותית. מצד שני, גם עבור אותו פוקימון, התמונות עשויות להשתנות בצורה משמעותית: תנוחה שונה של הפוקימון, זווית צילום שונה של הפוקימון באותה התמונה, רקע שונה, תאורה שונה, סגנון איור שונה (אנימה/קלפים/תלת-ממד/ציור ידני), ולעיתים גם חיתוך חלקי של דמות הפוקימון או פשוט רזולוציה נמוכה של התמונה.

במשימה הראשונה בפרויקט אנו מתמקדים בבעיית **Multiclass Classification** שבה לכל תמונה משויך **תיג יחיד** מתוך מספר רב של מחלקות (150 מחלקות). המטרה היא לאמן מודל למידה עמוקה שמקבל תמונה כקלט ומפיק הסתברות לכל אחת מהמחלקות, כך שהמחלקה בעלת ההסתברות הגבוהה ביותר תהיה שם הפוקימון הנכון. בניסוח פורמלי, מדובר בבעיה של למידה מפקחת (Supervised Learning) עם פונקציית הפסד המתאימה לסיווג רב מחלקתי כגון Cross-Entropy ומדד ביצוע מרכזי כגון מדד ה-Accuracy.

האתגר המרכזי במשימה הוא לגרום למודל ללמוד **מאפיינים ויזואליים עדינים** שמבדילים בין פוקימונים דומים (לדוגמה הבדלי צורה קטנים, פרופורציות, פרטים כמו זנב/קרניים/דפוסים), תוך שמירה על **יכולת הכללה (Generalization)** טובה לתמונות חדשות שלא נראו במהלך האימון. לכן נדרש איזון בין מודל בעל יכולת ייצוג גבוהה מספיק כדי להבחין בדקויות, לבין מנגנונים המפחיתים התאמת יתר (Overfitting) ומאפשרים למודל להתייחס לשונות סגנונית ומבנית בדאטה.

### **ניתוח נתונים ראשוני ואתגרים בפתרון הבעיה:**

#### **מקור הנתונים והמשימה**

הנתונים מאורגנים במבנה תיקיות `pokemon/<pokemon_name>/*.jpg...` כך שכל תיקייה מייצגת מחלקה (Label). מבנה זה מאפשר יצירה אוטומטית של Dataset לפי שמות תיקיות, ובפרט בניית מיפוי עקבי `id → label` המשמש את המודל באימון ובחזיון. בנוסף, פורמט זה מקל על בדיקות שעשינו במהלך חלק זה של הפרויקט כמו ספירת דוגמאות לכל מחלקה, והצגת דוגמאות ויזואליות לפי קטגוריות.

#### **האתגרים המרכזיים בחלק זה**

1. **חוסר איזון בין מחלקות:** מספר התמונות למחלקה אינו קבוע (בטווח של כ-215-45). מצב זה עלול לגרום לכך שהמודל ילמד טוב יותר מחלקות עם הרבה דוגמאות, ויתקשה במחלקות חלשות יותר. בנוסף, מדד Accuracy כללי עלול להסתיר ביצועים נמוכים במחלקות קטנות, ולכן בנוסף לניתוח עפ"י מדד ה-Accuracy בלבד הקפדנו גם לבצע ניתוח שגיאות באמצעות Confusion Matrix.
2. **Overfitting:** למרות שיש דאטה לא קטן, מדובר עדיין במספר תמונות מוגבל יחסית ביחס למספר מחלקות גבוה (150). רשתות CNN עלולות "לזכור" דוגמאות או להסתמך על מאפיינים לא רלוונטיים (למשל צבע רקע או סגנון איור נפוץ במחלקה), מה שמוריד ביצועים על תמונות חדשות. לכן יש צורך בשיטות לביצוע רגולריזציה כגון: Dropout, EarlyStopping וכדומה.
3. **דמיון חזותי:** פוקימונים שונים (ובפרט ההתפתחויות שלהם) יכולים להיראות דומים מאוד, וההבדלים ביניהם עשויים להיות קטנים (פרטים בגוף/ראש/זנב, גוונים, פרופורציות). בנוסף, כאשר התמונה חלקית או באיכות נמוכה, המודל עשוי להתבלבל בין מחלקות קרובות. לכן הקפדנו במהלך חלק זה של הפרויקט גם לבצע ניתוח זוגות בלבול נפוצים ולהציג דוגמאות שגויות שהמודל חוזה.

4. **תלות בתהליך הפרה-פרוססינג:** החלטות כמו גודל תמונה, נרמול התמונה, המרות צבע, וייצוב של משפיעות משמעותית על מהירות האימון, יציבותו והביצועים הסופיים.

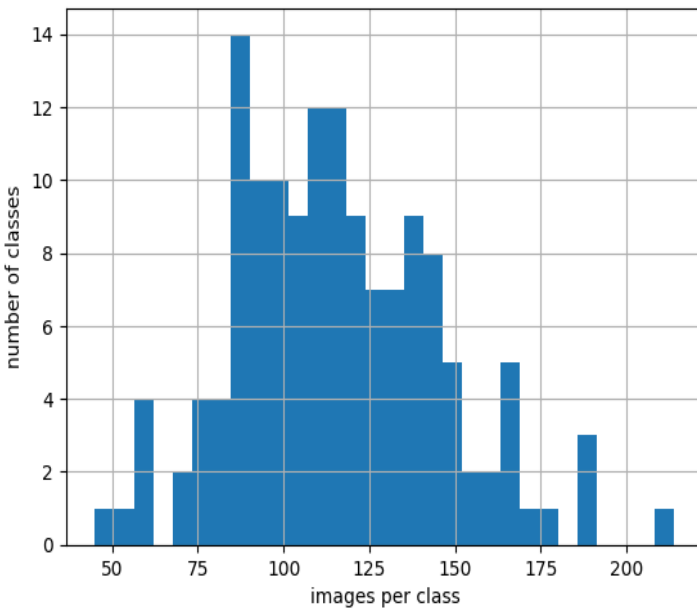
5. **רעש ויזואלי וסגנונות מגוונים:** תמונות עם רקעים עמוסים, טקסט/לוגו, חיתוכים לא מלאים או קונטרסט נמוך עשויות להקשות על המודל להתמקד בדמות עצמה. זהו גורם שמעלה קושי ומגדיל שונות תוך מחלקתית.

### ניתוח נתונים ראשוני

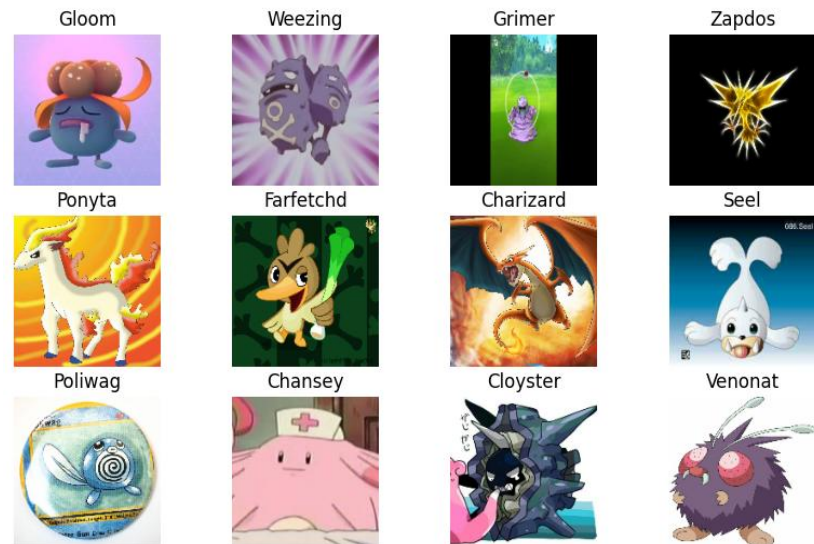
בוצע ניתוח ראשוני כדי להבין את הדאטה ולמנוע שגיאות אימון:

- **Audit לתמונות:** נבדקה תקינות של הקבצים, זוהו תמונות פגומות הוסרו כדי למנוע קריסות של המודל שנבנה בהמשך במהלך תהליך האימון שלו.
- **התפלגות מחלקות:** בנינו גרף היסטוגרמה של מספר התמונות לכל מחלקה. מצאנו שרוב המחלקות נעות סביב 80-150 תמונות, כאשר מחלקות חלשות יחסית נעות סביב 45-70 ומעט מחלקות חזקות מעל 180. תוצאה זו איששה בפנינו את הצורך בהתמודדות עם חוסר איזון למשל באמצעות Offline Augmentation למחלקות החלשות.
- **בדיקת דוגמאות:** במחברת הפיתוח, לצורך בדיקה שפיות הצגנו batches אקראיים מסט האימון בכדי לוודא שהתיוגים נכונים, שמיפוי המחלקה לשם הפוקימון בוצע בצורה נכונה, ושאינן טעינה שגויה של תמונות או היפוך צבעים ועיוותים חריגים בתמונות הפוקימונים.
- **בדיקות עקביות:** ביצענו חלוקה לסט אימון, סט וולידציה וסט מבחן באופן קבוע באמצעות seed ושמרנו את החלוקה כדי לאפשר השוואה הוגנת בין מודלים והרצות שונות.

Class count distribution



איור 1: התפלגות המחלקות השונות



איור 2: דוגמאות לתמונות של פוקימונים מתוך סט האימון

## הצגת תהליך החקירה ובניית הארכיטקטורה:

### חלוקת נתונים לשלושה סטים (Train - Validation - Test)

בחלק זה של הפרויקט ביצענו חלוקה לסטים של Train/Validation/Test ביחס 80% / 10% / 10% בצורה שהיא Stratified כלומר תוך כדי שמירה ככל האפשר על התפלגות המחלקות בכל קבוצה. מטרת החלוקה הייתה לאפשר:

- אימון על סט האימון בלבד.
  - בחירה וכיוונון של מודלים על סט הוולידציה והשוואת ארכיטקטורות שונות שנבחנו.
  - הערכה סופית והוגנת על סט המבחן שנשמר בצד עד סוף התהליך כדי למנוע מצב של overfitting.
- כדי לשמור על עקביות ושחזור של התוצאות, הקפדנו לשמור קבצי split קבועים (train.csv, val.csv, test.csv) ולהגדיר seed באופן קבוע. כך, כל הרצה תתבצע על אותה חלוקה של נתונים בדיוק, וההשוואה בין מודלים תישאר הוגנת.

### עיבוד מקדים (Preprocessing)

כיוון שמדובר בעבודה על מאגר נתונים גדול ולא פשוט, היינו זקוקים לייצר כאן סוג של pipeline יעיל שעובד בצורה סדורה:

- **Resize לכל התמונות לגודל של 160×160:** בחירה זו מאזנת בין זמן אימון לבין שמירה על מידע חזותי מספיק עבור זיהוי של 150 מחלקות.
- **המרה לפורמט של RGB:** המרה זו מאפשרת אחידות במספר הערוצים ומונעת בעיות בקבצים עם פורמטים שונים והכי חשוב, היא מאפשרת קלט סטנדרטי לכל ה-NN שנאמן בהמשך.
- **בדיקות תקינות:** לפני אימון בוצעו בדיקות קריאה לתמונות וזוהו תמונות בעייתיות, ולאחר מכן הוצגו batches אקראיים כדי לוודא שתוויות והטעינה תקינות.

### ביצוע אוגמנטציות

כדי להתמודד עם חוסר איזון במחלקות (מחלקות חלשות סביב 45-70 תמונות לעומת מחלקות חזקות), בוצע תהליך של אוגמנטציה:

- עבור מחלקות עם מעט דוגמאות נוצרו תמונות חדשות ונשמרו לוקאלית, עד ליעד של כ-120 תמונות עבור כל מחלקה מתוך 150 המחלקות בסט האימון.
- ביצוע האוגמנטציות נעשה באמצעות טרנספורמציות יחסית מתונות על התמונות כגון הזזה, סיבוב, זום קטן ועדין, שינוי ברמה הבהירות של התמונה וכו' וזאת כדי מצד אחד לשמר את זהות הפוקימון אך מהצד השני לייצר וריאציות שונות שלו בכדי להפוך את תהליך האימון ליעיל יותר.

היתרון של האוגמנטציות במקרה הזה הוא כפול:

1. איזון דאטה אמיתי ברמת התמונות והקבצים.
2. מבחינת שימוש ב-CPU הגישה הזאת מאפשרת שליטה טובה יותר ולא מכבידה יחסית על אימון הרשת בזמן אמת אצל הלקוח.

## Baseline CNN model - בניית מודל בסיסי

בנינו מודל Baseline כדי להבין עד כמה קל או קשה לאמן על הדאטה שלנו, ולוודא שהשיפורים בהמשך אכן משמעותיים ביחס למודל יחסית בסיסי וסטנדרטי.

### 1. Small CNN

רשת קונבולוציה בסיסית (Dense → Dropout → GAP → Pooling → Conv) שנועדה להיות מודל יחסית פשוט קל, מהיר ומובן. המטרה בבנייה של הרשת הזאת הייתה היא לקבל benchmark ראשוני ולהבחין האם הבעיה שלנו ניתנת לפתרון טוב ללא טכניקות מתקדמות כגון ביצוע Transfer learning.

### 2. מודל פשוט יותר להשוואה Logistic Regression על Features קפואים

החלטנו לבנות עוד סוג של baseline נוסף שמתבסס על רגרסיה לוגיסטית (אותה למדנו בקורס הקודם עם זוהר) המודל הזה מפריד בין חילוף תכונות לבין ה-classifier.

## מודלים מתקדמים ולמידת העברה (Transfer Learning)

לאחר יצירת מודל ה-baseline החלטנו ליישם את העיקרון של למידת העברה שנלמד בקורס על מודלים שאומנו מראש על הדאטה סט של imagenet כדי לנצל תכונות חזותיות כלליות ולהתאים אותן לדאטה של פוקימונים.

המודלים שנבדקו היו:

- **EfficientNetB0** - מודל מאוזן שמספק דיוק טוב ביחס למשאבים. האימון בוצע בשני שלבים:

1. **Frozen stage**: בסיס המודל היה קפוא הקלספייר שלו אומן על ידנו.

2. **Fine tune**: שחרור מהקפאה של חלק מהשכבות האחרונות ואימון בקצב למידה קטן יותר כדי לשפר התאמה לדאטה מבלי בעצם לשבור או להרוס את הידע הקודם שהמודל צבר.

- **MobileNetV3Large** - כפי שלמדנו בקורס, המודל של MobileNet הוא מודל יחסית קל ומהיר יותר, מתאים במיוחד לסביבה ללא משאבים גדולים (כמו טלפונים שאין להם GPU חזק), ההרצה של המודל הזה אפשרה לנו להשוות tradeoff בין זמן הריצה של המודל לבין הדיוק שלו.

## מלחמה ב-Overfitting וכיוונון Hyperparameters

במהלך ההרצות של המודלים החלטנו לנקוט במספר צעדים לשיפור הכללה של המודל ולהפחתת האוברפיטינג. בין הצעדים העיקריים שביצענו ניתן למנות את:

- הגדרה של callbacks לביצוע אימון יציב של המודל

- Model Checkpoint: שמירת המודל הטוב ביותר לפי סט הוולידציה.
- Early Stopping: עצירה מוקדמת של הרצת המודל כאשר אין שיפור, למניעת אוברפיטינג וחסכון בזמן.
- ReduceLROnPlateau: הורדת learning rate כאשר סט הוולידציה יחסית תקוע וזאת בכדי לאפשר שיפור של המודל.

- **כיוון של Hyperparameters**  
בדקו והשוונו ערכים עבור:

- שינוי של ה-Dropout
- שינוי של קצב הלמידה (Learning rate)
- שינוי מספר epochs בהתאם להתנהגות פונקציית ה-Loss ופונקציית ה-Accuracy

- **תיעוד והשוואה בין ההרצות השונות של המודלים**  
כל הרצה תועדה בטבלה אחידה שכללה: שם ריצה, זמן ריצה, best validation accuracy, test accuracy, או test רק פעם אחת בסוף לפי מודל נבחר. בנוסף עבור כל מודל יצרנו גרפים של ה-Loss ושל ה-Accuracy וזאת כדי לראות קיימת התכנסות לערך מסוים והאם יש פער בין Train ל-Validation.

### **הצגת התוצאות:**

- נבנתה טבלת השוואה בין הריצות של המודלים שכוללת דיוק של כל מודל.
- המודל המיטבי נבחר לפי **Validation Accuracy** ולאחר מכן הוערך על סט המבחן (Test set).
- לבסוף המודל המוביל היה Efficientnet\_Finetune שהציג תוצאות של בערך 86% דיוק על סט המבחן ו-84% דיוק על סט הוולידציה.

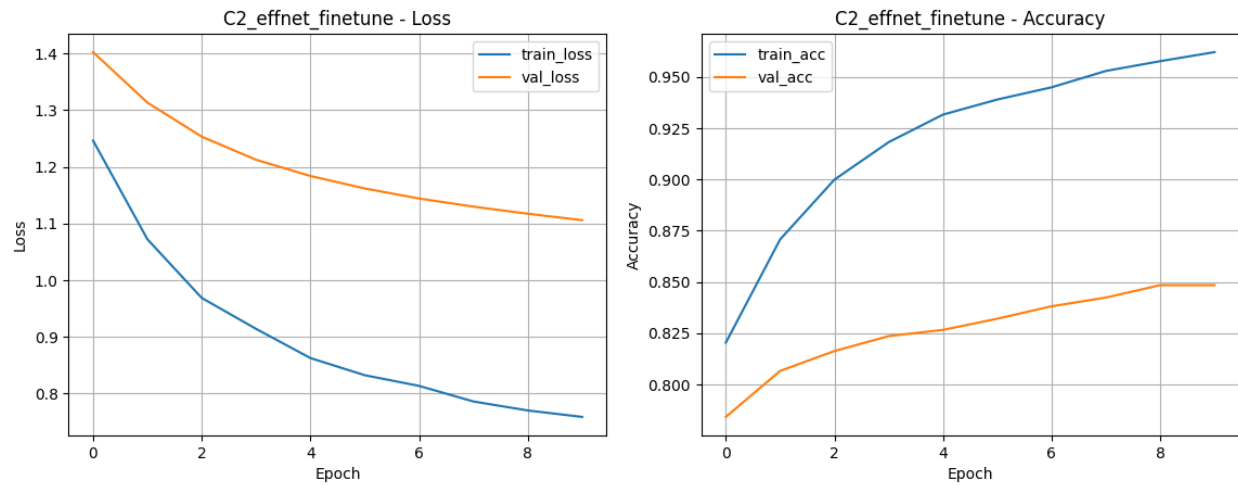
○ **Best Validation Accuracy: 0.848**

○ **Test Accuracy: 0.864**

	run name	best_val_acc	test_acc
0	Efficientnet_Finetune	0.848	0.864
1	Efficientnet_Frozen	0.840	0.863
2	MobileNet_Frozen	0.827	0.859
3	Lr_on_MobileNet_features	0.795	0.828
4	Small CNN	0.293	0.297

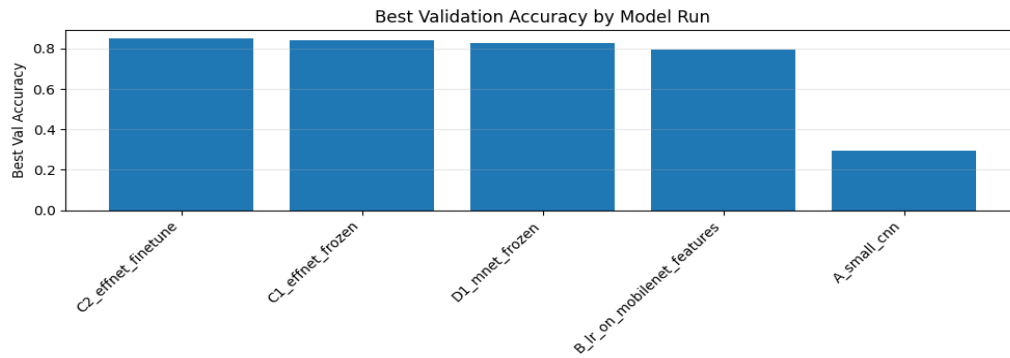
## גרפים וויזואליזציות

גרפי של פונקציית ההפסד (Loss) ופונקציית הדיוק (Accuracy) של המודל המדויק ביותר (Efficientnet\_Finetune):

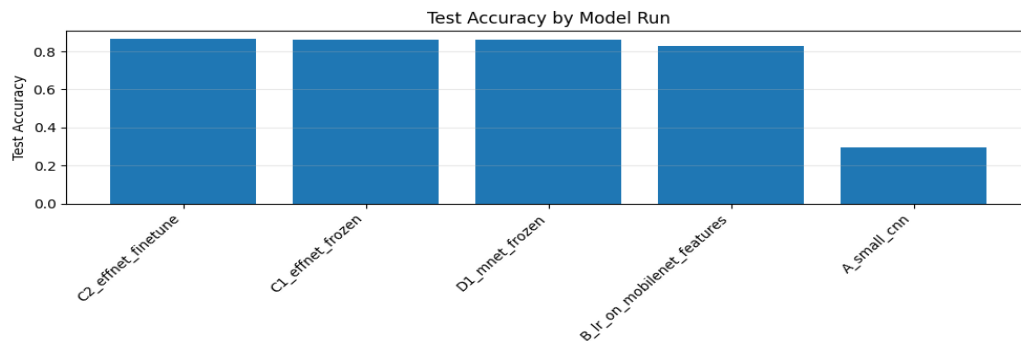


איור 3: תוצאות המודל המדויק ביותר Efficientnet\_Finetune

גרפי השוואה בין מודלים:



איור 4: השוואה של אחוזי דיוק בין המודלים על סט הוולידציה



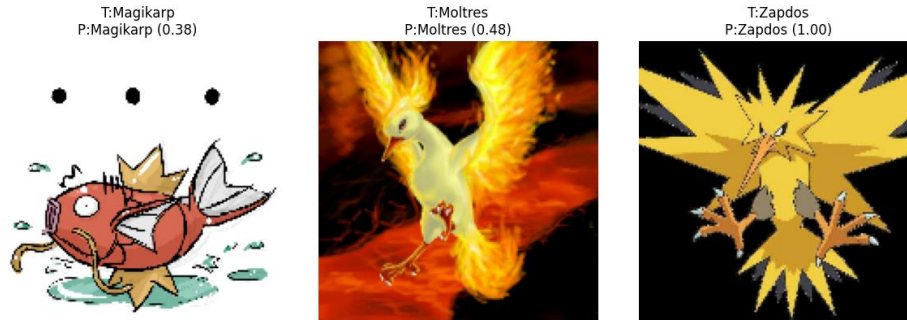
איור 5: השוואה של אחוזי דיוק בין המודלים על סט המבחן



## דוגמאות הצלחות וכישלונות

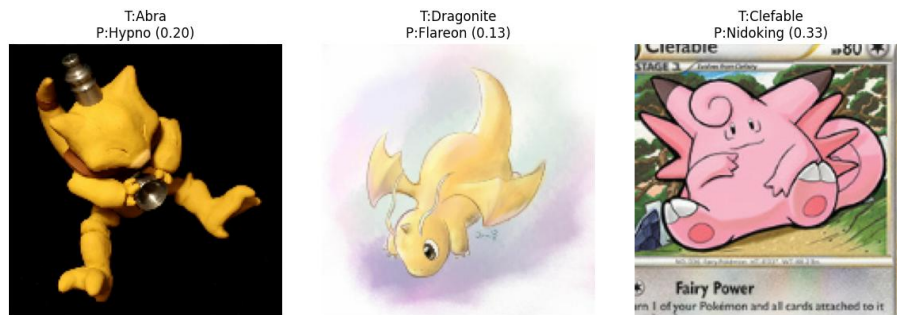
כעת לאחר שסיימנו את הניסויים שלנו ואת שלב הערכת התוצאות, בחרנו להתקדם עם המודל של Efficientnet\_Finetune והתחלנו לבחון אותו על דוגמאות שונות של פוקימונים מסט המבחן.

### • 3 תמונות שהמודל זיהה באופן נכון:



איור 6: תמונות של פוקימונים שהמודל של Efficientnet\_Finetune זיהה באופן נכון

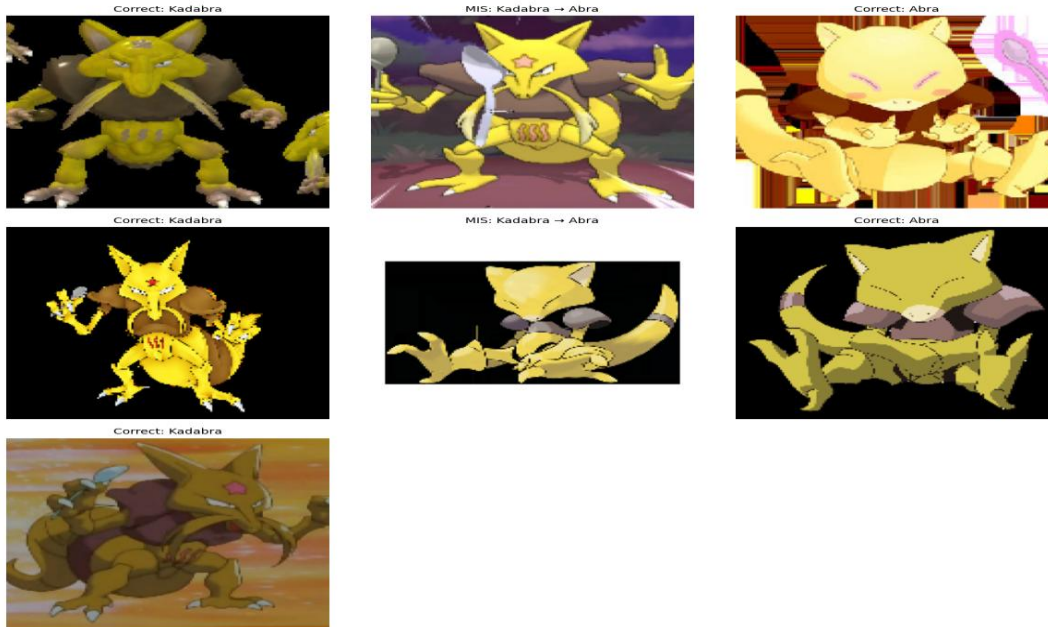
### • 3 תמונות שהמודל זיהה באופן שגוי:



איור 7: תמונות של פוקימונים שהמודל של Efficientnet\_Finetune זיהה באופן שגוי

בשלב הבא הבחנו שלמרות שהמודל יחסית מדויק, הוא לעיתים עדיין מתבלבל בין תמונות דומות של פוקימונים, בעיקר התפתחויות שונות של אותו הפוקימון.

Confusion Pair Gallery: Kadabra → Abra



איור 8: דוגמה לבלבול נפוץ של המודל *Efficientnet\_Finetune*, הוא מתבלבל בין הפוקימון *Abra* לפוקימון *Kadabra* שהם התפתחות אחד של השני

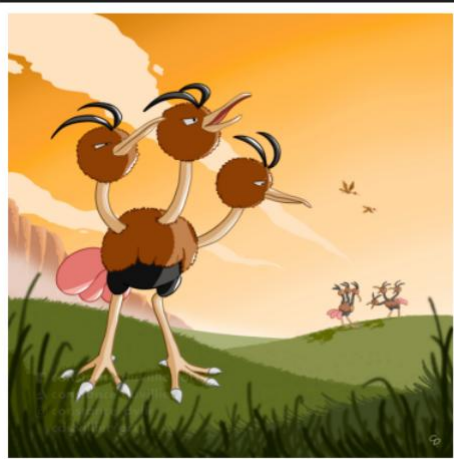
### שמירת המודל ובדיקת יכולת הכללה שלו על תמונות חדשות מהאינטרנט שהוא מעולם לא ראה

שמרנו את המודל המאומן כך שניתן יהיה להשתמש בו גם לאחר סיום האימון. לאחר מכן ביצענו **בדיקת הכללה (Generalization)** על תמונות חדשות שלא הופיעו בדאטה סט המקורי שעליו עבדנו מאתר Kaggle. הורדנו מספר תמונות של פוקימונים מהאינטרנט, העברנו אותן את אותו תהליך preprocessing במסגרתו עשינו לכל תמונה **resize** לגודל של  $160 \times 160$  והמרה לפורמט RGB והרצנו את המודל של *Efficientnet\_Finetune* עליהן (המודל שנתן את התוצאה הטובה ביותר).

התוצאות הראו שהמודל מצליח **לזהות בצורה טובה** גם תמונות חדשות ממקורות חיצוניים, למרות הבדלים אפשריים בסגנון ציור, איכות תמונה ורקע. ממצא זה מחזק את הטענה שלנו שהמודל לא רק שינן את הדאטה סט, אלא גם למד מאפיינים חזותיים כלליים שמאפשרים לו לבצע סיווג מדויק גם על דוגמאות חדשות של פוקימונים.

Top predictions:

Dodrio	0.883
Doduo	0.050
Fearow	0.009
Zubat	0.004
Primeape	0.003



איור 11: תמונה חדשה של הפוקימון דודריו שלקחנו מסצנה בסדרה

Top predictions:

Bulbasaur	0.846
Ivysaur	0.062
Jigglypuff	0.025
Flareon	0.007
Clefairy	0.005



איור 10: תמונה חדשה של הפוקימון בלבזאור שלקחנו מגוגל.

Top predictions:

Snorlax	0.632
Farfetchd	0.173
Primeape	0.023
Machop	0.023
Mewtwo	0.021



איור 13: תמונה של בובה של הפוקימון סנורלקס, גם כאן המודל מציג ביצועים יחסית טובים.

Top predictions:

Blastoise	0.822
Wartortle	0.112
Rhydon	0.022
Machop	0.005
Squirtle	0.005



איור 12: תמונה חדשה של הפוקימון בלסטויז שלקחנו מתוך סצנה בסרט של פוקימון

## **חלק ב' של הפרויקט**

### **רקע לבעיה השנייה - בעיית Multi-Label Classification :**

סיווג תמונות (Image Classification) הוא אחד התחומים המרכזיים ביותר בעולמות של למידה עמוקה ורשתות נוירונים. עם זאת, בעוד שבעיות רבות של סיווג תמונה מנוסחות כבעיות Multi-Class (כל תמונה יכולה להיות שייכת למחלקה אחת מתוך מספר מחלקות), קיימות בעיות שבהן כל תמונה יכולה להשתייך ליותר ממחלקה אחת בו-זמנית. בעיות אלו מוגדרות כבעיות Multi-Label Classification והן נפוצות בעולם האמיתי בתחומים כמו תיוג תמונות (למשל "אדם", "כלב", "רכב" באותה תמונה) או בניתוח תמונות רפואיות (תסמינים מרובים באותו צילום רפואי). בפרויקט זה אנו מתמקדים בבעיה מסוג זה: חיזוי סוגי פוקימון (Pokemon Types) מתוך תמונה, כאשר לכל פוקימון יכול להיות **סוג ראשי אחד** ולעיתים גם **סוג משני**, כלומר עד שתי תוויות (labels) עבור אותה תמונה.

ביקום של Pokémon קיימים 18 סוגים שונים של פוקימונים (למשל: Fire, Water, Grass וכו') והסוגים של פוקימון משפיעים על תכונותיו, יחסי יתרון או חולשה בקרבות מול פוקימונים אחרים, ועל האופן שבו שחקנים בונים אסטרטגיה שלהם במשחק. מטרת הפרויקט היא לפתח מודל למידה עמוקה המקבל כקלט תמונת פוקימון ומחזיר כפלט את הסוגים המתאימים לו. בשונה מעבודות רבות אשר מתייחסות לסיווג סוג אחד בלבד (לרוב הסוג הראשי), אנו מנסים לבצע חיזוי מלא יותר: גם של סוג ראשי וגם של סוג משני.

במונחי למידה סטטיסטית, המשמעות היא שהמודל שלנו אינו נדרש לבחור מחלקה יחידה נכונה, אלא להפיק הסתברות לכל אחד מ-18 הסוגים האפשריים, כאשר יותר מסוג אחד יכול להיות נכון.

ניסוח כזה של הבעיה מחייב התאמה הן של ארכיטקטורת המודל והן של פונקציית ההפסד שלנו (Loss Function) והמדדים להערכה, משום שהנחות בסיסיות של SoftMax אינן תקפות כאשר קיימת אפשרות לשתי תוויות נכונות בו-זמנית עבור פוקימון יחיד.

### **אתגרים עיקריים של המשימה הזאת:**

#### **1. ביצוע משימה של Multi-Label (1-2 סוגים אפשריים לכל תמונה)**

כאשר לכל תמונה עשויים להיות שני סוגים של תוויות, שימוש בפונקציית אקטיבציה מסוג SoftMax ובפונקציית הפסד מסוג Categorical Cross-Entropy אינו מתאים לפתרון הבעיה. פונקציית SoftMax מניחה שהסתברויות של כלל המחלקות מסתכמות ל-100% ושיש מחלקה אחת נכונה. בפועל, במקרה שלנו מודל טוב צריך לתת הסתברות גבוהה למספר סוגים במקביל (למשל הסתברות גבוהה ל-Water ו-Flying עבור פוקימון שהוא גם מסוג מים וגם מסוג מעופף). לכן, פתרון נפוץ הוא שימוש באקטיבציה מסוג Sigmoid לכל Label ובפונקציית הפסד מסוג Binary Cross-Entropy, או לחלופין לעבוד עם מודל שהוא Two-Head שמנבא סוג ראשי וסוג משני בנפרד.

#### **2. חוסר איזון בנתונים שלנו (Class Imbalance)**

אחד האתגרים המשמעותיים שהתמודדנו איתם הוא חוסר איזון בין הסוגים השונים של הפוקימונים - יש סוגים שכיחים מאוד כמו Water\Normal ולעומתם סוגים נדירים מאוד כמו Ice\Dragon\Fairy. חוסר איזון כזה גורם לכך שהמודל עלול להצליח לכאורה במשימתו אם ילמד להעדיף סוגים שכיחים, אך בפועל ייכשל בזיהוי סוגים נדירים - בדיוק אלו שלרוב חשוב לנו לזהות נכון. בעיות מסוג זה דורשות שימוש בגישות כמו Weighted Loss, Oversampling.

### 3. שונות חזותית בין הפוקימונים ותנאי צילום שונים

תמונות פוקימונים עשויות להגיע מסגנונות שונים (תמונת בסגנון Artwork, תמונות בסגנון משחק, רקעים שונים, צבעוניות שונה, זוויות שונות). שינוי הרקע או התאורה או הרזולוציה עלולה להשפיע על המודל ולהקשות עליו לבצע הכללה נכונה. גם אם הפוקימון עצמו מוצג בצורה יחסית נקייה, עדיין יש שונות בין תמונות של אותו פוקימון ובין דורות שונים.

### 4. קביעת Threshold נכון להמרת ההסתברויות ל-Labels

במודל שמבצע משימה של Multi-Label Classification המודל מחזיר הסתברות לכל סוג, אך בסוף צריך להחליט אילו תוויות נבחרות ומוצגות כפלט. זה נעשה בדרך כלל באמצעות Threshold (למשל 0.3/0.5), אך אין ערך אחד שתמיד מתאים לכל הסוגים של הפוקימונים. סף נמוך מדי ייצור הרבה False Positives וסף גבוה מדי יפספס תוויות נכונות. לעיתים נכון יותר להשתמש ב-Threshold שונה לכל מחלקה או לבחור Top-K במיוחד כאשר ידוע שלכל תמונות של פוקימון Label יחיד או שתי Labels בלבד. במקרה שלנו בחרנו לעבוד עם  $Top-k=2$ .

### עבודות קודמות שנעשו בנושא:

בתחום סיווג פוקימונים קיימות עבודות ופרויקטים קודמים שמדגימים שימוש ברשתות CNN ובלמידת העברה (Transfer Learning) בכדי לזהות מאפיינים חזותיים של פוקימונים ולהגיע לדיוק גבוה יחסית. עם זאת, חלק ניכר מהפתרונות הקיימים כרגע מתמקדים בבעיה פשוטה יותר מהבעיה שלנו - הם מבצעים ניבוי של ה-Primary type של אותו הפוקימון או שהם מבצעים ניבוי של השם של אותו פוקימון לפי התמונה שלו (כמו שהצלחנו לעשות בהצלחה יחסית רבה בחלק הראשון של הפרויקט). פתרונות כאלה נהנים ממבנה פשוט יותר של פונקציית הפסד ופונקציית דיוק, ולעיתים גם ממערכי נתונים מאוזנים יותר או כאלה שמכילים פחות וריאציות של אותה תמונה.

החלק הנוכחי של הפרויקט רוצה לקחת את הידע שנצבר בנושא עד כה צעד אחד קדימה בכך שהוא מתמודד במפורש האתגר של Multi Label.

### ניתוח נתונים ראשוני:

בשלב הניתוח הראשוני ביצענו היכרות עם מאגר הנתונים שלקחנו מ-Kaggle, הן ברמת הקלט הוויזואלי (התמונות) והן ברמת התוויות עצמן בקובץ ה-CSV מטרת שלב זה הייתה להבין את מאפייני הדאטה, לזהות בעיות פוטנציאליות (כגון חוסר איזון, רשומות חסרות או תמונות פגומות), ומכאן לגזור החלטות בנוגע להמשך תהליך העבודה.

קובץ ה-CSV המקורי שמגיע מ-Kaggle מכיל 809 רשומות ובנוי בצורה הבאה: לכל אחד מ-809 הפוקימונים יש עמודה שמציינת את השם שלו (Name), עמודה שמציינת את ה-Type1 שלו, עמודה שמציינת את ה-Type2 שלו ועמודה שמציינת את ה-Evolution שלו (פחות רלוונטי לפרויקט שלנו, לכן הסרנו אותה)

	Name	Type1	Type2	Evolution
0	bulbasaur	Grass	Poison	ivysaur
1	ivysaur	Grass	Poison	venusaur
2	venusaur	Grass	Poison	NaN
3	charmander	Fire	NaN	charmeleon
4	charmeleon	Fire	NaN	charizard

איור 14: נתוני קובץ ה-CSV

בשלב הבא עברנו לדאטה סט שמכיל את 809 התמונות של הפוקימונים וחקרנו קצת עליו. בשלב הראשון הצגנו כמה תמונות לדוגמה כדי לחוש את הדאטה ובשלב הבא וחקרנו על הסוג שלהם (תמונות PNG או JPEG ברזולוציה לא קבועה). ביצענו resize לתמונות (לגודל 120 פיקסלים על 120 פיקסלים) כדי שהמודל יתאמן על תמונות בגודל אחיד.

בשלב הבא יצרנו עמודה חדשה בקובץ ה-CSV שנקראת Type ומכילה את הסוגים של הפוקימון (יכול להיות סוג אחד בלבד ויכול להיות שני סוגים). המטרה הייתה שעמודת ה-Type תחליף את העמודות של Type1 ו-Type2.

	Name	Type1	Type2	Evolution	Type
0	bulbasaur	Grass	Poison	ivysaur	[Grass, Poison]
1	ivysaur	Grass	Poison	venusaur	[Grass, Poison]
2	venusaur	Grass	Poison	NaN	[Grass, Poison]
3	charmander	Fire	NaN	charmeleon	[Fire]
4	charmeleon	Fire	NaN	charizard	[Fire]

איור 15: יצירת טבלה חדשה שמכילה עמודה בשם Type

לאחר מכן ביצענו התאמה בין כל תמונה של פוקימון ל-Label של ה-Type שמייצג אותו. בתמונה שלמטה רואים את התוצר הראשוני שקיבלנו, כעת הגענו למצב שלכל פוקימון יש את התווית שמייצגת את הסוגים שלו.

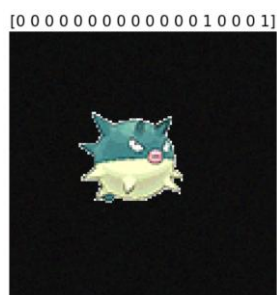


איור 16: התאמה בין תמונות הפוקימון ל-Label שלו

לאחר מכן עברנו לביצוע encoding ל-Label וזאת כדי שלמודל יהיה קלט מנורמל לעבוד איתו. בחרנו לבצע encoding בשיטה של MultiLabelBinarizer עליה חקרנו וגילינו שהיא מתאימה למשימות מסוג multi-label בהן יש הרבה classes שונים (במקרה שלנו 18 מחלקות שונות).

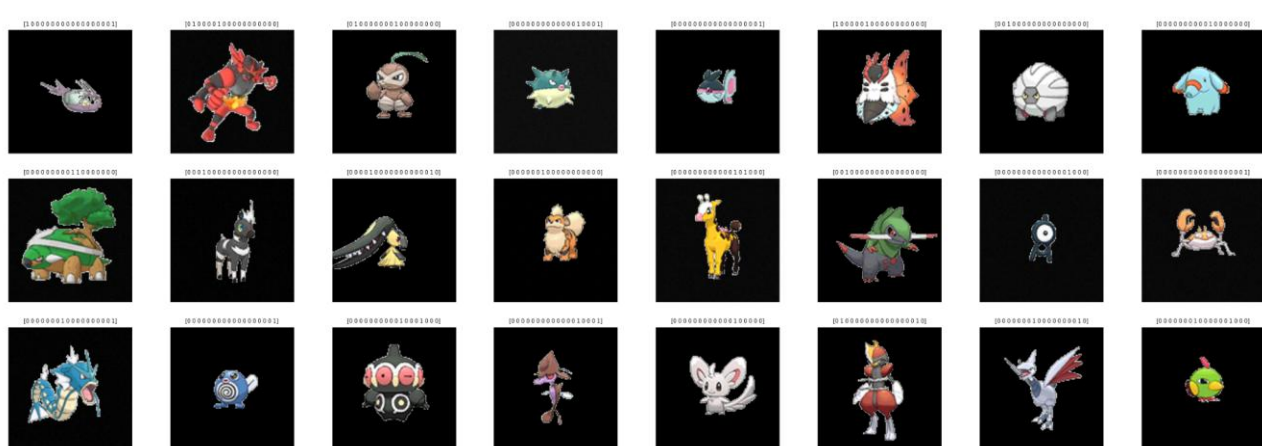


בתהליך ה-encoding הזה כל Label יהיה מיוצג ע"י ווקטור באורך 18 (כמספר המחלקות שלנו), כאשר הערך 0 בווקטור יצביע על כך שהפוקימון לא שייך ל-type הנוכחי והערך 1 יציע שהפוקימון כן שייך ל-type הנוכחי. בתמונה למטה ניתן לראות פוקימון שהסוג שלו הוא מים ורעל ואת ה-Label המקודד שלו.



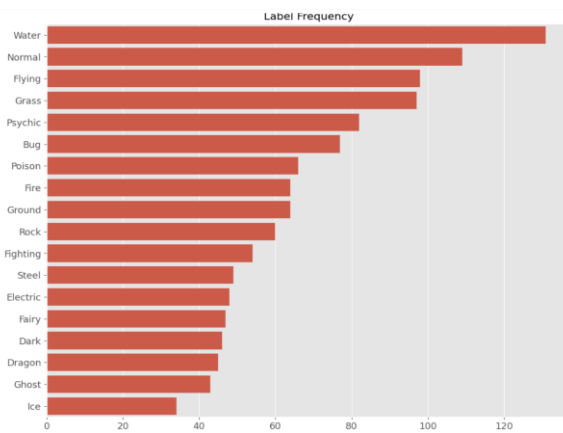
איור 17: ביצוע תהליך ה-Encoding

לבסוף הגענו לתוצר הבא:



איור 18: דוגמאות לדאטה סט הסופי המכיל תמונה ו-Label מקודד עבור כל פוקימון

בשלב הבא עברנו לנתח את המחלקות עצמן - הצגנו התפלגות של ה-labels השונים של הפוקימונים בדאטה שלנו וראינו שההתפלגות לא אחידה. ניתן לראות שיש יותר מ-120 פוקימונים שהסוג שלהם הוא מים ואילו רק 25 פוקימונים שהסוג שלהם הוא קרח ו-40 פוקימונים שהסוג שלהם הוא רוח. כלומר הדאטה שלנו לא מאוזן.



איור 19: התפלגות הפוקימונים לפי ה-Label שלהם

**אוגמנטציות -** לאחר הניתוח הראשוני של הדאטה זיהינו כי כמות התמונות לאימון המודל יחסית נמוכה (809 תמונות בלבד וזאת עוד לפני שביצענו חלוקה לסט אימון, סט ולידציה וסט מבחן). כלומר כמות התמונות הזמינה לאימון המודלים היא יחסית מוגבלת ומאחר שמדובר במשימת סיווג תמונות, קיימת סכנה של התאמת יתר (overfitting) - מצב שבו המודל לומד מאפיינים ספציפיים מאוד לתמונות האימון ואינו מצליח להכליל לדוגמאות חדשות. על מנת להתמודד עם בעיות אלו, החלטנו לבצע Data Augmentation כפי שלמדנו בקורס בשיעור על התמודדות על דאטה קטן במודל CNN.

האוגמנטציות נועדו להשיג מספר מטרות מרכזיות: בראש ובראשונה הגדלת נפח הדאטה – כלומר יצירת וריאציות חדשות של כל תמונה קיימת מבלי לאסוף נתונים נוספים. בנוסף שיפור יכולת ההכללה של המודל - כאשר אנחנו מבצעים אוגמנטציות על התמונות המודל נאלץ ללמוד מאפיינים חזותיים כלליים (צורה, צבע, קווי מתאר) ולא להסתמך על זיהוי פרטים מקריים.

האוגמנטציות נבחרו בקפידה כך שיתאימו לאופי התמונות (פוקימונים עם דמות מרכזית ורקע פשוט יחסית), ויימנעו משינויים שעלולים לשנות את משמעות התמונה או את סוג הפוקימון. להלן סוגי האוגמנטציות שבחרנו לבצע וההיגיון שעומד מאחוריהן:

- הזזה וביצוע זום עדין - בוצעה הרחבה של גבולות התמונה באמצעות padding ולאחר מכן חיתוך אקראי וחזרה לגודל המקורי של התמונה. פעולה זו מדמה שינוי קל במיקום הפוקימון בתוך התמונה, ומאלצת את המודל להיות פחות רגיש למיקום המדויק של האובייקט. ההיגיון מאחורי הפעולה הוא שבפועל, פוקימון יכול להופיע מעט שמאלה, ימינה או במרכז התמונה, והסוג שלו אינו תלוי במיקום.
- שינוי בהירות וניגודיות - בחרנו לבצע שינויי בהירות וניגודיות קלים, בטווחים שמוגדרים כשמרניים. ההיגיון מאחורי המהלך היה שמאפייני צבע ותאורה עשויים להשתנות בין תמונות (או עקב עיבוד שונה), אך סוג הפוקימון נשאר זהה. אוגמנטציה זו מונעת מהמודל שלנו להינעל על ערכי צבע מדויקים.
- שינוי קל של ה-Saturation - צבעים הם מאפיין חשוב בסיווג סוגי פוקימונים, אך שינוי קל ברוויה של הצבעים (Saturation) לא אמור לשנות את הסוג. לכן בוצעו שינויים מינוריים בלבד, כדי לא לפגוע במשמעות הסמנטית של הצבע.
- הוספת רעש גאומטרי קל - לתמונות הוסף רעש אקראי בעוצמה נמוכה. הרציונל מאחורי בחירה זו היה שרעש קל מאלץ את המודל ללמוד דפוסים יציבים ומשמעותיים ולא מאפשר לו להסתמך על פיקסלים בודדים או פרטים חדים מדי.
- אוגמנטציות שנמענו מהן במכוון - במהלך הפרויקט נמנענו מאוגמנטציות אגרסיביות מדי כגון: טשטוש חזק (Blur) שינויי גוון משמעותיים וחיתוכים אגרסיביים שעלולים להסתיר חלקים מרכזיים מהדמות של הפוקימון. אוגמנטציות אלו עלולות לפגוע במידע קריטי לסיווג סוג הפוקימון, במיוחד כאשר צבע וצורה הם מאפיינים דומיננטיים בתהליך הזיהוי.

## **הצגת תהליך החקירה ובניית הארכיטקטורה:**

### **מודל הבסיס:**

מודל הבסיס שנבנה בפרויקט הוא רשת CNN המיועדת לחילוץ מאפיינים חזותיים מתמונות של פוקימונים ולהמרתם לייצוג שמאפשר סיווג. הארכיטקטורה שבנינו כוללת חמישה בלוקים קונבולוציוניים, כאשר בכל בלוק מופעלת שכבת קונבולוציה עם פילטר בגודל  $3 \times 3$  ופונקציית אקטיבציה מסוג ReLU מספר הפילטרים עולה בהדרגה לאורך הרשת ( $16 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow 150$ ), כך שהשכבות הראשונות לומדות מאפיינים יחסית בסיסיים של הפוקימונים כמו קצוות ומרקמים, והשכבות העמוקות יותר לומדות דפוסים מורכבים של פוקימונים ברמת אבסטרקציה גבוהה יותר הקשורים לצורה ולמבנה של הדמות. לאחר כל קונבולוציה ביצענו Batch Normalization לצורך ייצוב ונרמול הפלט של השכבה ושיפור יציבות ומהירות האימון, ולאחר מכן ביצענו Max Pooling בגודל  $2 \times 2$  כלומר down sampling של ה-feature maps שנוצרו. בסיום החלק של ה-Feature extraction, שיטחנו את

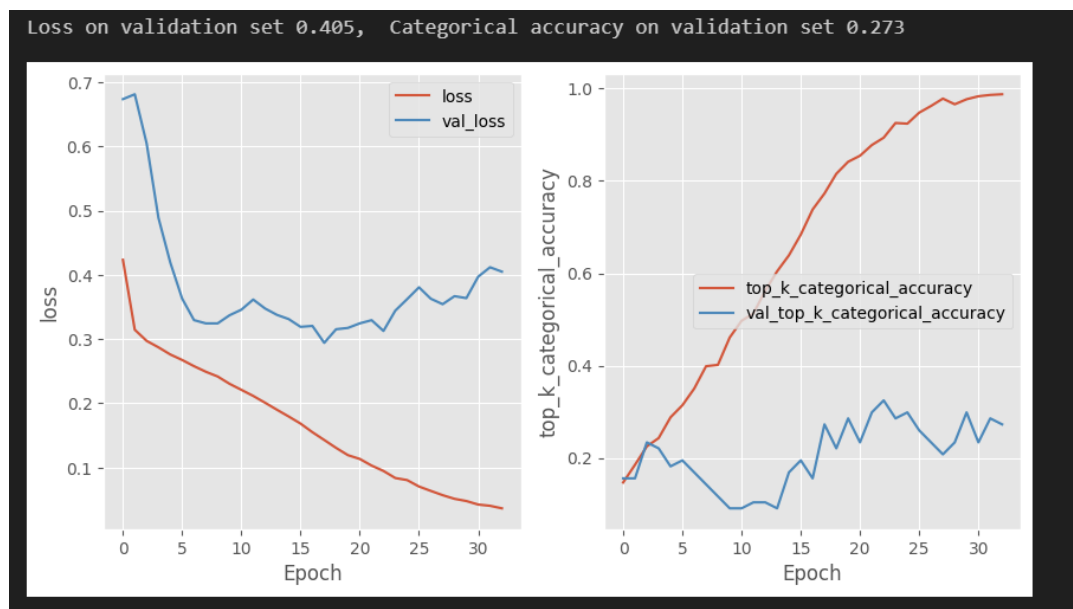


הפלט לווקטור חד מימדי (ביצוע flatten) והעברנו אותו לשכבה Dense(64, ReLU). לבסוף מופיעה שכבת פלט בגודל N\_LABELS המפיקה ציון לכל קטגוריה אפשרית של פוקימון ומהווה את בסיס ההחלטה של המודל בסיווג.

### מודל שמבוסס על Transfer Learning ממודל של MobileNet:

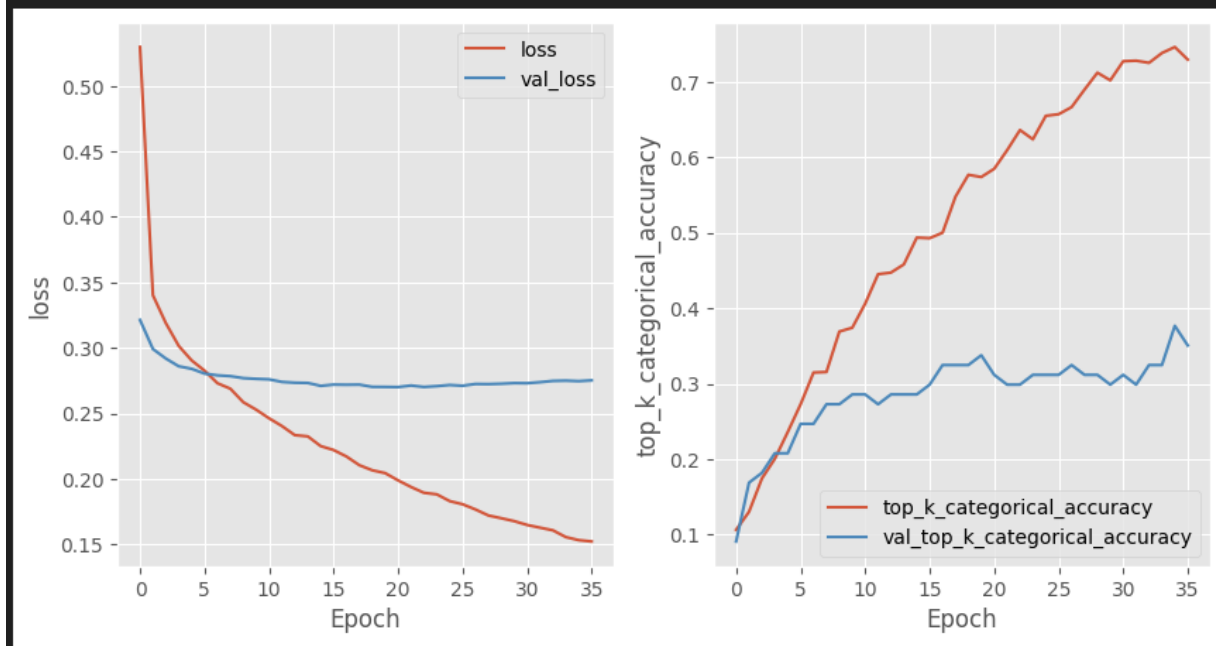
המודל מבוסס על גישת Transfer Learning באמצעות MobileNet שמשמש במקרה הזה כמודל בסיס מאומן מראש. MobileNet מספק ייצוג חזותי עשיר המבוסס על ידע שנלמד ממאגר גדול כגון ImageNet ובכך מאפשר השגת הכללה טובה יותר גם כאשר כמות הנתונים בדאטה היא יחסית מוגבלת. מעל ה-backbone של מובייל נט הוספו קלאספייר ייעודי הכולל שכבת GlobalAveragePooling2D אשר מבצעת ממוצע מרחבי על ה-feature maps. לאחר מכן נוספה שכבה Dense(128, ReLU) לצורך התאמת הייצוג הכללי של MobileNet למשימת הסיווג הספציפית איתה אנחנו מתמודדים, וכן שכבת Dropout(0.2) שמבצעת רגולריזציה לשיפור הכללה. לבסוף, שכבת פלט בגודל N\_LABELS מפיקה ציונים עבור כל קטגוריה אפשרית של פוקימונים, כאשר הערכת הביצועים מתבצעת על בסיס דירוג התחזיות (Top-K) כלומר בדיקה האם התווית הנכונה מופיעה בין K התחזיות בעלות ההסתברות הגבוהה ביותר (במקרה שלנו  $K=2$ ).

### הצגת התוצאות:



איור 20: תוצאות של מודל הבסיס

Loss on test set 0.275, Categorical accuracy on val set 0.351

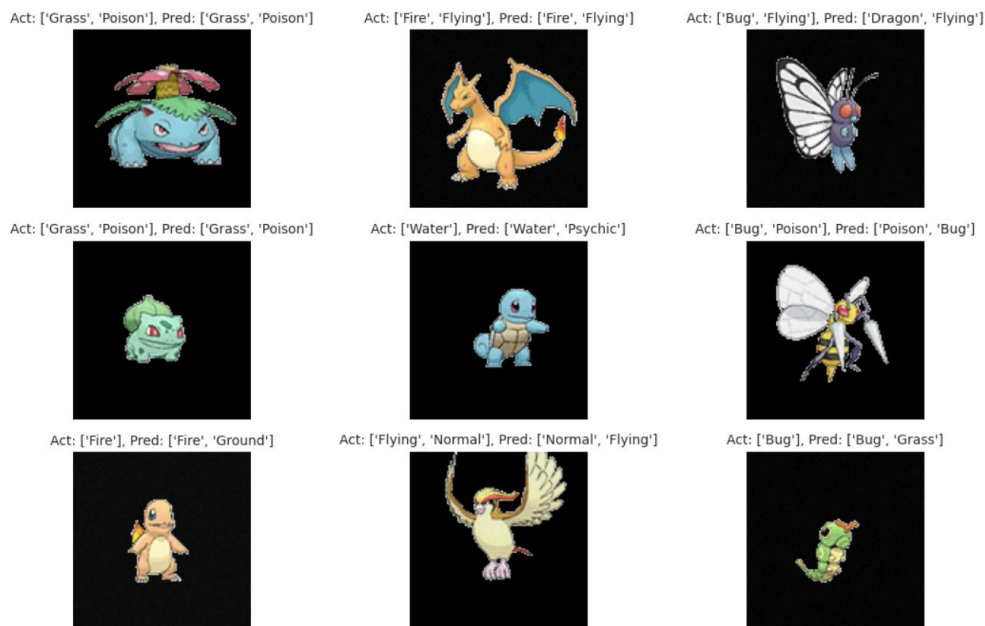


איור 21: תוצאות של מודל המבוסס על MobileNet

מתוצאות עולה כי קיים פער ביצועים מובהק בין מודל ה-CNN הבסיסי שבנינו לבין המודל המבוסס על MobileNet. במודל הבסיסי נצפית ירידה עקבית ב training loss בעקבות העדכון של משקלי המודל, אך validation loss מתחיל לעלות לאחר מספר epochs. תופעה זו, יחד עם העובדה שה- Top-K categorical accuracy על סט הוולידציה נותר נמוך יחסית (בסביבות 0.25-0.30), מצביעה על נטייה של המודל הבסיסי לבצע overfitting ועל הכללה מוגבלת לנתונים חדשים שהמודל לא ראה במהלך האימון. לעומת זאת, המודל המבוסס על MobileNet מציג התנהגות יציבה ומשופרת יותר: ה- validation loss נמוך ועקבי לאורך האימון, וה- Top-K categorical accuracy על סט הוולידציה וסט המבחן מגיע לערכים גבוהים יותר (כ-0.35).

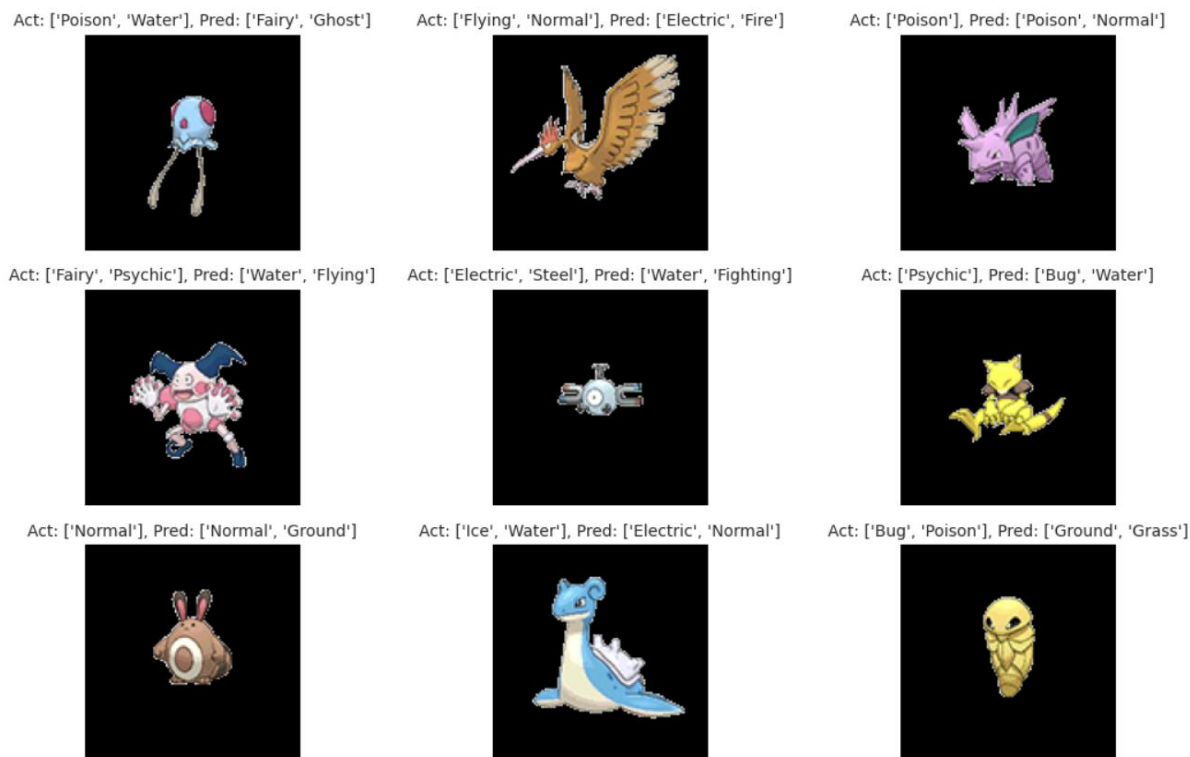
תוצאות אלו מצביעות על כך שהשימוש ב-Transfer Learning מאפשר למודל לנצל ייצוגים חזותיים כלליים שנלמדו מראש, ובכך לשפר את יכולת הדירוג של הקטגוריות הנכונות גם כאשר אין התאמה מושלמת בניבוי הראשון. בסך הכול, הערכה באמצעות Top-K categorical accuracy מדגישה כי MobileNet מצליח לדרג בצורה טובה יותר את סוגי הפוקימונים הרלוונטיים בין התחזיות המובילות, ומהווה פתרון יעיל ומכליל יותר בהשוואה למודל הבסיסי. אם זאת חשוב לציין שעדיין קיים מקום לשיפור, זאת כנראה בגלל כמות הדאטה הקטנה והקושי הממשי לחזות באופן מדויק את 2 הסוגים של הפוקימונים.

## בחינה של המודל הנבחר transfer learning mobile net על תמונות מסט האימון:



איור 22: תוצאות של MobileNet בלמידת העברה על סט האימון

## בחינה של המודל הנבחר transfer learning mobile net על תמונות מסט האימון:



איור 22: תוצאות של MobileNet בלמידת העברה על סט המבחן

## בחינה של המודל על תמונות חדשות שהוא מעולם לא ראה, במקרה הזה תמונות של פוקימונים מדור 8 ומדור 9:

הסוגים האמיתיים של הפוקימונים הם: Grookey מסוג Grass, Sprigatito הוא מסוג Grass ו-Fuecoco הוא מסוג Fire.

Grookey (Gen8)  
Fairy: 16.6%  
Normal: 10.7%  
Ghost: 10.1%  
Fire: 9.9%  
Grass: 9.4%  
Water: 9.2%  
Rock: 8.3%  
Ground: 7.0%  
Fighting: 5.5%  
Poison: 5.5%  
Psychic: 5.4%  
Flying: 4.8%  
Bug: 4.3%  
Electric: 3.3%  
Steel: 1.4%  
Dragon: 1.2%  
Ice: 1.2%  
Dark: 1.2%



Sprigatito (Gen9)  
Grass: 21.9%  
Bug: 15.8%  
Psychic: 13.7%  
Fairy: 12.8%  
Ghost: 5.5%  
Water: 4.2%  
Rock: 3.2%  
Ice: 2.6%  
Fire: 2.6%  
Flying: 2.4%  
Normal: 2.4%  
Electric: 2.4%  
Fighting: 2.3%  
Poison: 2.3%  
Ground: 1.1%  
Steel: 1.0%  
Dark: 0.9%  
Dragon: 0.6%



Fuecoco (Gen9)  
Fire: 43.3%  
Psychic: 17.4%  
Ground: 16.2%  
Fighting: 11.7%  
Rock: 7.9%  
Grass: 6.3%  
Electric: 5.9%  
Flying: 4.9%  
Steel: 4.8%  
Water: 4.6%  
Normal: 4.1%  
Ice: 2.9%  
Ghost: 2.4%  
Bug: 1.8%  
Dragon: 1.6%  
Poison: 1.4%  
Fairy: 0.6%  
Dark: 0.6%



פרדיקציות שהמודל הנבחר מבצע על תמונות פוקימונים מדור 8 ודור 9, תמונות שהוא מעולם לא ראה.

## סיכום ומסקנות:

בפרויקט זה ביצענו שתי משימות סיווג תמונות של פוקימונים **Multiclass Classification** לזיהוי 150 פוקימונים מדור 1, ומשימה של **Multi-Label Classification** לחיזוי 2 סוגים של פוקימון מדורות 1-7 כאשר לכל פוקימון עד שני סוגים.

שתי המשימות הדגישו את ההבדל בין סיווג "קלאסי" של multiclass לבין סיווג יותר מתקדם וקשה של multilabel, ואת ההשפעה של גודל הדאטה, חוסר איזון בין מחלקות, ורגולריזציה על יכולת ההכללה של כל מודל שבנינו במהלך הפרויקט.

בחלק הראשון עבדנו עם דאטה גדול יחסית (17,000 תמונות בערך), אך עם אתגרים של דמיון חזותי בין מחלקות, שונות גבוהה בתוך מחלקה וחוסר איזון בכמות התמונות. ראינו ש-**Transfer Learning** עדיף משמעותית על מודל הבייסליין. המודל הטוב ביותר היה **EfficientNet** עם Fine-Tuning שהגיע לאחוזי דיוק של כמעט 85% כאשר רוב הטעויות שלו היו בין פוקימונים דומים מאוד (למשל התפתחויות אחד של השני).

בחלק השני הדאטה היה קטן מאוד (809 תמונות בלבד, לרוב תמונה אחת לכל פוקימון), ולכן האתגרים המרכזיים איתם התמודדנו היו אוברפיטינג וחוסר איזון בין המחלקות בין הסוגים השונים של הפוקימונים. ביצענו השוואה בין baseline CNN לבין **MobileNet** ב-**Transfer Learning** וראינו שיפור עקבי: מדד Top-K Categorical Accuracy (K=2) הגיע ל-35%, מספר גבוה יותר מהבייסליין, אך עדיין מוגבל בגלל כמות הדאטה והקושי לחזות שני Labels מדוגמה אחת בלבד.

לסיכום, הפרויקט מראה שלמידת העברה משפרת ביצועים בשתי המשימות, אך מדגיש את הקושי וההבדל בין משימת ה-Multi-Label שהיא יותר מורכבת משמעותית מביצוע משימה של Multi-Class.

## נספחים:

<https://www.kaggle.com/code/amikhshibu/transfer-learning-tf-inceptionv3-val-cc-82>

<https://towardsdatascience.com/multi-label-classification-of-pokemon-types-with-tensorflow-8217a38038a6/>

<https://www.kaggle.com/code/allenlu112220/pokemon-type-classification>

<https://www.kaggle.com/code/josutk/pokemon-type-identify>

תוצה על הקריאה

ניה, 138 וסיטל