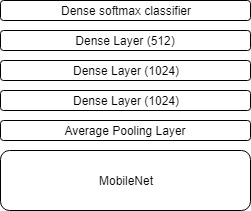
* 1. תאור המערכת\אלגוריתם
     1. מודל CNN
        1. ארכיטקטורה

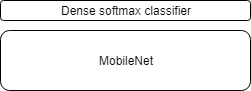
במסגרת עיצוב המערכת, ניסינו להימנע ככל הניתן מהמצאת הגלגל מחדש, ורצינו להסתמך ככל האפשר על מערכות וידע קיימים. לאור זאת, בכדי לבנות את מודל הCNN שלנו, בחרנו להיעזר בטכניקת Transfer Learning. מבין שלל המודלים הניתנים לשימוש בframework של Keras, בחרנו את המודל של MobileNet, שכן מודל זה פותח כאשר הוא מיועד להיות משובץ בתוך מערכות mobile ו-embedded, ולכן מכיל מספר שכבות מועט יחסית, ובנוי משכבות depthwise convolution, אשר מצמצמות את סיבוכיות החישובים בכל שכבה.

כדי להמשיך וליהנות מיתרונות עבודה קודמת שנעשתה על ידי אחרים, השתמשנו במשקולות המודל MobileNet אשר נלמדו על ה-dataset של ImageNet. משקולות אלו מהווים עבורנו נקודת התחלה מצוינת, שכן כך אנו מקבלים את השכבות הנמוכות במודל, אשר מזהות מאפיינים, פינות וקווים בסיסיים, כאשר הן כבר אומנו על מעל ל-14 מיליון תמונת, הרבה מעבר ל-dataset המצומצם שיש בידינו, המכיל 4326 תמונת בלבד.

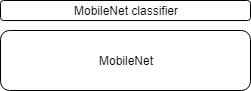
כמובן שהיה צורך להחליף את שכבת ה-classifier הקיימת במודל (איור 1) בכזאת המתאימה לבעיה שאנו מנסים לפתור. כזכור, ב-ImageNet ישנם 1000 סיווגים, בעוד בבעיה שלנו ישנם 5 בלבד. תחילה הוספנו ישירות מעל שכבות ה-CNN שכבת Dense אחת עם פונקציית אקטיביציה softmax (איור 2), אך לאחר מספר ניסויים שביצענו, הגענו לקונפיגורציה שבה מעל לשכבות הקונבולוציה של MobileNet, ישנה שכבת Average Pooling, מעליה שתיה שכבות Dense בעלות 1024 נוירונים, שכבת Dense נוספת בת 512 נוירונים, ולבסוף שכבת ה-classifier, בת 5 נוירונים, עם פונקציית אקטיבציה softmax (איור 3).



איור 3 – ארכיטקטורה סופית



איור 1 - מודל MobileNet



איור 2 - מודל MobileNet עם classifier

1. אימון המודל

שלב האימון דרש התאמות והתייחסות לעובדה שהמודל משלב שכבות מאומנות ושכבות שאינן מאומנות. בכדי לא לפגוע בשכבות התחתונות, אשר כאמור מזהות מאפיינים בסיסיים אותם תחום הבעיה שלנו חולק עם תחום הבעיה של ImageNet, הקפאנו את משקולות 20 השכבות הראשונות במודל. בחרנו את ה-optimizer שלנו להיות RMSprop, עם learning rate נמוך יחסית של , כדי שלא לשנות יותר מדי משקולות קיימים במודל.

כאמור, הdataset שלנו מצומצם מאוד, ובכדי להגדיל את מספר הדוגמאות שהשכבות העליונות של המודל שלנו רואות, חיפשנו דרכים לבצע data augmentation. בין השיטות בהן השתמשנו, נעזרנו בהפיכה אופקית, זום, שינויים ברוחב ובגובה התמונה, וסיבוב התמונה בכיוון השעון.

במסגרת הניסויים השונים שביצענו בשלב בחירת המודל המתאים לבעיה, ניסינו לבחור את מספר הepochים המתאים ביותר, אשר ישלב כמובן בין validation accuracy גבוה, ובין זמן אימון מינימאלי. צמצום הסיבוכיות ורידוד המודל ע"י שימוש ב-MobileNet כמודל עליו התבססנו הובילו, ככל הנראה, גם למשך אימון קצר יותר. במדידות אמפיריות שערכנו, עולה כי epoch בודד במודל על גבי מודל MobileNet לקח כשמינית מן הזמן שלקח לנו לבצע במודל עמוק יותר, לדוגמה Inception V3. כמו כן, הצלחנו להשיג validation accuracy של 87% לאחר 8 epoch-ים בלבד. ניסיון להגדיל את מספר ה-epoch-ים לא הראה שיפור משמעותי בביצועי המודל, והחל ממספר epoch-ים כלשהו, באופן צפוי התחלנו לראות ירידה ב-validation accuracy.

1. ממשק משתמש

הבעיה אותה המערכת מנסה לפתור היא בעיה קלה להבנה עבור בני אדם, כזאת שהקלט והפלט הצפויים עבורה פשוטים וברורים. בפיתוח ממשק המשתמש, בחרנו לשמור ככל הניתן על פשטות זאת, כדי לאפשר למספר רב ככל האפשר של משתמשים להיעזר ולהשתמש במערכת, עם דרישות מינמאליות ככל הניתן של ידע קודם בשימוש במחשבים.

ממשק המשתמש מחולק גיאוגרפית לשני חלקים, בהתאם לשתי אפשרויות השימוש במערכת. החלק הראשון, המופיע גיאוגרפית בחלקו העליון של ממשק המשתמש, עוסק בטעינת נתונים, ודרישה מהמודל לפסק סיווג עבור הנתונים. בנוסף לציון נתיב לתיקייה שבה נמצאות התמונות אותן ברצון המשתמש לסווג, נדרש המשתמש לספק גם נתיב לקובץ h5 המכיל את המודל השלם (כלומר, ארכיטקטורה ומשקולות).

חלקה השני של המערכת, המופיע גיאוגרפית בתחתית ממשק המשתמש, עוסק בהצגת תוצאות הסיווג ועיבוד נתונים. בטבלה, אשר תופסת את מרבית שטח הממשק, יוצגו שמות הקבצים אשר נטענו מתוך הנתיב אשר אותו סיפק המשתמש, ולצידם יוצגו הסיווגים אשר התקבלו על ידי המודל. מצד שמאל של חלקו התחתון של הממשק, הוספנו אפשרות לסנן את התוצאות המוצגות בטבלה, ע"י בחירת הסיווג. כלומר, אם בחרנו Daisy, יופיעו רק תמונות אשר קיבלו את הסיווג Daisy על ידי המודל. ניתן לבחור כל שילוב שרירותי של סיווגים.