זיהוי מחלות בעלי שעועית בעזרת כלים של למידת עמוקה

מגישה: אביטל אקרמן 208933606

תאריך הגשה : 20.02.2022

מבוא

לשעועיות מגוון רב של תכונות חיוביות המשפיעות על בריאותנו :

1. שעועית היא מקור נהדר לחלבון מהצומח. בניגוד למקורות חלבון מן החי השעועית לא מכילה שומן רווי שמזיק לבריאות שלנו. כשאנו אוכלים שעועית אנו מקבלים את היתרונות של חלבון ללא הסיכון למחלות לב שנגרם מאכילה של שומן רווי.
2. הן מקור נהדר לסיבים תזונתיים שעוזרים להרגשה של שובע אחרי ארוחה, עושים רגולציה לרמות הסוכר בדם ומאפשרים למערכת העיכול לעבוד כראוי.
3. בנוסף השעועיות מכילות מגוון רחב של ויטמינים ומינרלים אשר חיוניים לנו.

זאת ועוד שעועית מהווה במדינות רבות באמריקה ואפריקה 15% מהצריכה הקלורית היומית ו 36% מצריכת החלבון היומית.[1]

בגלל כל היתרונות האלו אנחנו נרצה לשמור על בריאות השעועית על ידי זיהוי מהיר של גורם המחלה ומתן טיפול מתאים ובכך להציל את השעועית אם אפשר, ואם לא אז לדעת איזה שעועית היא חולה ולא מתאימה למאכל אדם. זיהוי מוקדם של גורם המחלה וטיפול בו יכול להפחית אובדן מזון בשדות ובכך להגדיל את כמות האוכל הזמינה לאדם. בעולם בו יש כ 820 מיליון אנשים רעבים ויש אובדן של כ 10-40% של פירות , ירקות ודגנים בשדה בשלב הקטיף, אשר חלק מן האובדן נובע ממחלות. לכן הצורך בדרך זיהוי מהימנה ומהירה הוא גדול ומשמעותי.

בפרויקט הזה נזהה 2 מחלות שיש בשעועית :

1. Angular leaf spot- זוהי מחלה בפטרייתית הנגרמת מ פטרייה בשם Phaeoisariopsis griseola. המחלה יכולה להדביק כל צמח ממשפחת cucurbit שהשעועיות הן חלק ממנה. זו מחלה נפוצה מאוד בשעועית ויכולה לגרום לאובדן של עד 70% מהיבול .העלים מפתחים נקודות קטנות ועגולות בצבע חום עם הילה צהובה . הנקודות האלו מתייבשות ונופלות ויוצרות עלים עם חורים. כשמחלה מדביקה את הצמח היא ממשיכה לתוך הפרי ומדביקה את הזרעים שלו . [1]
2. leaf Rust-מחלה זו נגרמת מפטרייה Uromyces appendiculatus ומשפיעה על מינים שונים של שעועית. היא משפיעה על העלים , גבעולים ותרמילים. המחלה יכולה לגרום לאובדן של עד 30% מהיבול. סימני המחלה הם נקודות קטנות וצהובות שמדביקות את הרקמה העליונה של העלה . ככל שהמחלה ממשיכה הנקודות הופכות לשחורות וגדולות ויכולות לגרום לעלה ליפול.[2]

אלו מחלות שנראות מאוד דומה לעין לא מיומנת. לכן יש צורך לדעת להבדיל בניהן בצורה טובה . אחת השיטות האפשרויות לזיהוי היא על ידי מודלים של למידת מכונה . ניתן לצלם את עלי השעועית ולהעלות את התמונות למודל שנבנה מראש ואומן על ידי אלפי תמונות לצורך זיהוי מצבו של העלה . מודלים כאלו משמשים כיום בהמון תחומים שונים ומניבים תוצאות טובות ,מהירות ואמינות. רמת האמינות גבוה מאוד , שיכולה להתקבל רק על ידי אדם מומחה בתחום . על ידי מודלים כאלו כל אדם (בעל שדה לא מיומן במקרה שלנו) יכול לזהות את מצבה של

השעועית ללא מאמץ רב . בנוסף ישנו ההתרון המאפשר לבדוק כמות גבוהה של תמונות בזמן קצר מאוד ומאפשר לכסות מספר רב של יבולים בזמן קצר מאוד.

הדאטאסט

את הדאטאבייס לקחתי מ TensorFlow זהו דטאסט מוכר [3].

בדאטאטסט הזה ישנם תמונות של עלי שעועית משלושה חלקים : בריאים , חולים ב lea rust או חולים ב Angular leaf spot. התמונות הן בפורמט jpg והינן בגודל של 500 על 500 עם שלושה ערוצים : אדום , ירוק וכחול .

<u>הדאטאסט מגיע מחולק ל 3 חלקים :</u>

1. Train- זו התיקייה הגדולה ביותר שעליה המודל מתאמן. על התמונות האלו המודל מעדכן בכל epoch את המשקולות שלו כדי לשפר את הביצועים שלו . יש בה 1,034 תמונות .
2. Validation – התיקייה שמכילה תמונות שיעזרו להעריך את ביצועי המודל. על תמונות אלה המודל לא יתעדכן, אלא רק ישמשו כדי לדעת עד כמה המודל טוב. יש בה 133 תמונות .
3. Test – זו התיקייה שמכילה תמונות חדשות שלא משמשות לאימון. הן מדמות תמונות חדשות שהמודל יקבל בעתיד, ומהוות למעשה סט מבחן עבור המודל. יש בה 128 תמונות.

דוגמא לתמונות בדאטאסט:

Angular leaf spot

Healthy

Leaf rust

לקחתי תמונה של עלה שעועית שאינה נמצאת בדאטאסט הזה כדי לבדוק את התיוג שהיא מקבלת במודלים הנבחרים . [4]

בנוסף לכך לקחתי תמונה של עלה מלפפון החולה ב Angular leaf spot כדי לראות האם המודל יכול לזהות גורם מחלה המבוטא בצורה דומה בעלה שונה . [5]

תוצאות

ראשית כדי להתמודד עם הבעיה השתמשתי ב fastai2 כפי שלמדנו בשיעור . ניסיתי להשתמש בשני גדלים של מודל resnet וגדלים שונים של התמונה . כדי לראות האם בהוספה של מידע

השיפור בדיוק הזיהוי יהיה גבוה יותר . בהוספה של מידע יש אומנם הוספה של פרטים המאפשרים את הזיהוי אך יש גם trade off למהירות.

בפרויקט השתמשתי ב google colab , עם הרצה על ה GPU.

הוספתי לכל קוד גם שמירה אוטומטית של המודל הטוב ביותר ועצירה של האימון כאשר התוצאות לא משתפרות יותר. מכיוון שמטרת המודל היא בסופו של דבר לזהות תמונות שהוא עוד לא "ראה", העצירה תהיה לא לפי אחוז ההצלחה, אלא לפי כמות ה loss, שהוא המדד לכמות השגיאה. סביר להניח ש loss נמוך על סט ה validate (שעליו אין עדכון משקלים), משמעותו שהמודל יודע להכליל בצורה טובה גם על תמונות שהוא לא "ראה" בזמן האימון.

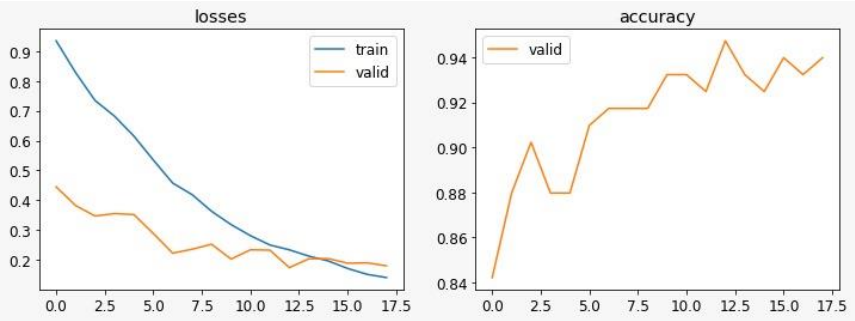
המודלים:

- <u>Resnet18, עם שינוי גודל התמונה ל 128x128</u>

epoch	train_loss	valid_loss	accuracy	time
0	1.373877	0.535956	0.834586	00:11

Better model found at epoch 0 with valid_loss value: 0.5359559655189514.

epoch	train_loss	valid_loss	accuracy	time
0	0.934499	0.444953	0.842105	00:11
1	0.828165	0.382144	0.879699	00:11
2	0.734433	0.347111	0.902256	00:11
3	0.682075	0.355147	0.879699	00:11
4	0.615126	0.351829	0.879699	00:11
5	0.534725	0.288527	0.909774	00:11
6	0.457375	0.222306	0.917293	00:11
7	0.418511	0.235741	0.917293	00:11
8	0.362862	0.252370	0.917293	00:11
9	0.318310	0.202598	0.932331	00:11
10	0.280992	0.233692	0.932331	00:11
11	0.249863	0.232616	0.924812	00:11
12	0.233636	0.173920	0.947368	00:11
13	0.212575	0.203536	0.932331	00:11
14	0.196446	0.204296	0.924812	00:11
15	0.171201	0.188676	0.939850	00:11
16	0.151465	0.190136	0.932331	00:11
17	0.140571	0.179994	0.939850	00:11



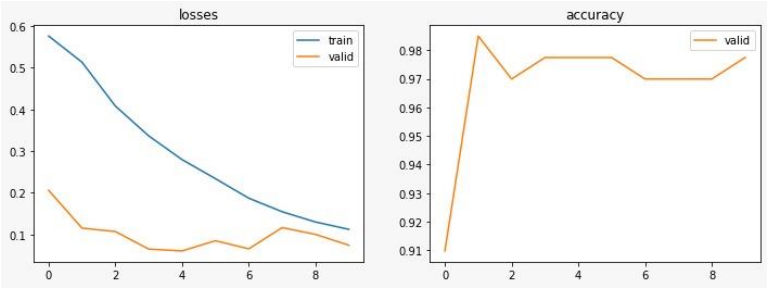
ניתן לראות שמודל זה הגיע 94.7% דיוק ב validate ב epoch 12, והוא גם ה epoch עם ה loss הנמוך ביותר ב validate.עם זמן הרצה של 11 שניות לכל epoch. אפשר לראות שגם בהקטנת התמונה, מודל resnet18 מגיע לביצועים טובים.

- <u>Resnet18 ללא הקטנת התמונה</u>

epoch	train_loss	valid_loss	accuracy	time
0	1.442204	0.456607	0.834586	00:50

Better model found at epoch 0 with valid_loss value: 0.45660704374313354.

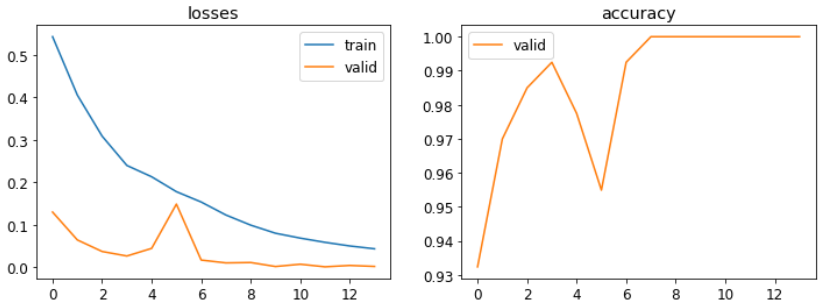
epoch	train_loss	valid_loss	accuracy	time
0	0.575928	0.206021	0.909774	01:05
1	0.513249	0.115351	0.984962	01:05
2	0.408099	0.107046	0.969925	01:05
3	0.336474	0.065028	0.977444	01:05
4	0.279604	0.060524	0.977444	01:05
5	0.233875	0.085142	0.977444	01:05
6	0.187099	0.065520	0.969925	01:05
7	0.154552	0.116498	0.969925	01:05
8	0.129937	0.100231	0.969925	01:05
9	0.112371	0.074294	0.977444	01:05



במודל זה אין ספק שזמן הריצה היה ארוך יותר, אך עדיין סביר ומהיר יחסית. ניתן לראות שהמודל הגיע לתוצאות טובות הרבה יותר, של 98.4%, אך עם זמן ריצה של 1:05 לכל epoch.

- <u>Resnet34 ללא הקטנת התמונה</u>

epoch	train_loss	valid_loss	accuracy	time
0	0.542263	0.129948	0.932331	02:00
1	0.405038	0.064504	0.969925	02:00
2	0.308542	0.037341	0.984962	02:00
3	0.239589	0.026722	0.992481	02:00
4	0.212993	0.044645	0.977444	01:59
5	0.177947	0.148729	0.954887	01:59
6	0.153632	0.017184	0.992481	01:59
7	0.123078	0.010543	1.000000	01:59
8	0.099284	0.011419	1.000000	01:59
9	0.080260	0.001998	1.000000	01:59
10	0.068770	0.007396	1.000000	01:59
11	0.058856	0.001276	1.000000	01:59
12	0.050320	0.004513	1.000000	01:59
13	0.043651	0.002198	1.000000	01:59



ניתן לראות שהמודל הגיע לדיוק מושלם על סט ה validate עם ערך loss נמוך מאוד. כנראה שמודל זה מכליל בצורה מאוד טובה. אך עם זמן ריצה של 1:59 לכל epoch.

אין ספק שמודל זה היה טוב מאוד, והאימון שלו היה ארוך יותר מהאימון של resnet18, אבל עדיין לא ארוך מדי, ולכן ויתרתי על בדיקת המודל של resnet34 עם הקטנת התמונה. בנוסף, לאחר מספר הרצות, Google Colab הגיע למגבלת הריצה שלו ולכן נאלצתי להשהות את העבודה.

אחר כך ניסיתי לבנות מודל ב keras במטרה להגיע לתוצאות כמה שיותר טובות. כמובן שמודל resnet שאומן על מיליוני תמונות יגיע לביצועים טובים יותר ממודל פשוט שנבנה במסגרת פרויקט מסכם לקורס, אבל אשאף להגיע לתוצאות כמה שיותר טובות.

מצאתי מודל מוכן ומתאים לדאטאסט הזה באינטרנט וניסיתי לשפר אותו.[6]

באתר בו המודל מתואר, הוא מגיע ל 91% דיוק על סט ה validate לאחר 70 epochים, והקוד לא לוקח את המודל שנתן את התוצאות הטובות ביותר. לכן הרצתי את הקוד בעצמי כדי לראות את ביצועי המודל לאורך התהליך.

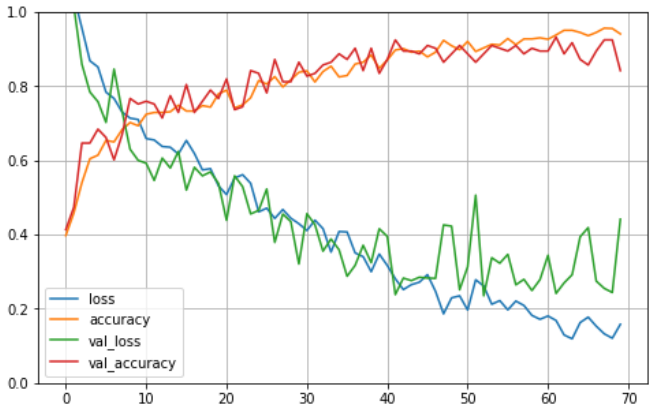
המודל הגיע ב epoch 42 ל 93% דיוק על סט ה validate עם loss נמוך. לכן זו נקודת הייחוס של שיפור המודל. עם זמן הרצה של כ53 שניות ל epoch.

<u>המודל:</u>

1. שכבת קלט בגודל 500x500x3 (כגודל התמונה, ו 3 הצבעים)
2. הקטנת התמונה ל 125x125
3. שכבת קונבולוציה של 3x3 עם 64 פילטרים + שכבת Max Pool
4. שכבת קונבולוציה של 3x3 עם 64 פילטרים + שכבת Max Pool
5. שכבת קונבולוציה של 3x3 עם 128 פילטרים + שכבת Max Pool
6. שכבת קונבולוציה של 3x3 עם 128 פילטרים + שכבת Max Pool
7. שכבת קונבולוציה של 3x3 עם 128 פילטרים + שכבת Max Pool
8. שכבת Fully Connected בגודל 256
9. שכבת Fully Connected בגודל 128

10. שכבת פלט בגודל 3

כאשר לכל שכבה (למעט שכבת הפלט), מוגדרת פונקציית האקטיבציה ReLU והורדת כמות ה epochים ל 50 .

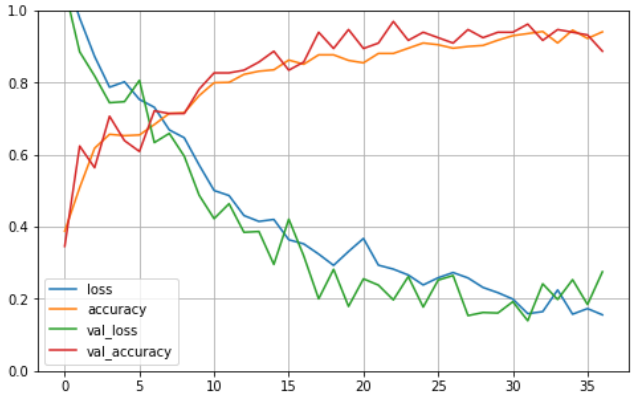


<u>ניסיון 1 :</u>

המודל המקורי הקטין כל תמונה לגודל של 125x125, מה שסביר להניח מוביל לאיבוד רב של מידע. לכן התחלתי מלהסיר את ההקטנה של התמונות. התוצאה הייתה קריסה של ה Google Colab, על כך שאין מספיק זיכרון כדי להריץ את המודל.

<u>ניסיון 2 :</u>

זהה למודל המקורי, אך עם הקטנת כל תמונה לגודל של 250x250.



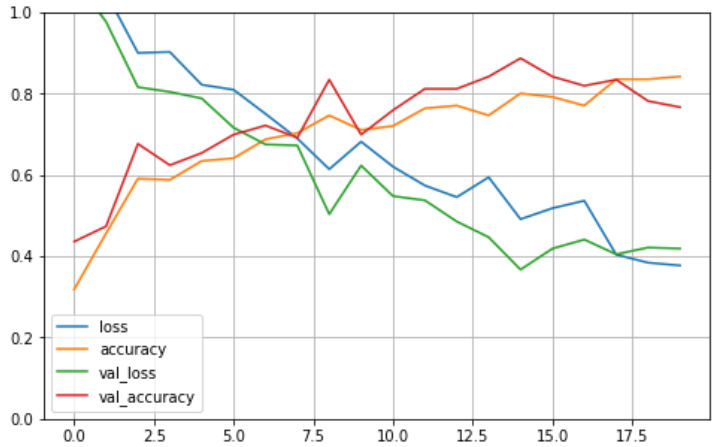
מודל זה הגיע לדיוק של 96%, ב epoch 32, ובו גם ערך ה loss היה הנמוך ביותר. ע"י שינוי הגודל של התמונה לגודל גדול יותר, התוצאות של המודל המקורי, השתפרו ב 3%. עם זמן הרצה של 50 שניות לepoch.

<u>ניסיון 3 :</u>

זהה לניסיון 2, אך עם שינוי פונקציית האקטיבציה ל LReLU עם $\alpha = 0.05$ בכל שכבה, ועם הפחתת שכבת ה Fully Connected בגודל 128. המודל הסופי :

1. שכבת קלט בגודל 500x500x3 (כגודל התמונה, ו 3 הצבעים)

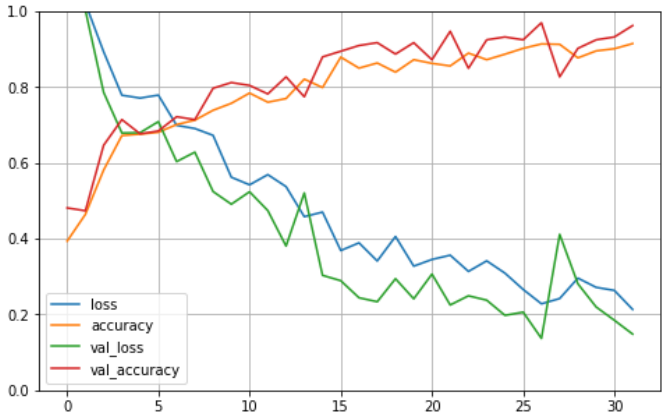
2. הקטנת התמונה ל 250x250
3. שכבת קונבולוציה של 3x3 עם 64 פילטרים + שכבת Max Pool
4. שכבת קונבולוציה של 3x3 עם 64 פילטרים + שכבת Max Pool
5. שכבת קונבולוציה של 3x3 עם 128 פילטרים + שכבת Max Pool
6. שכבת קונבולוציה של 3x3 עם 128 פילטרים + שכבת Max Pool
7. שכבת קונבולוציה של 3x3 עם 128 פילטרים + שכבת Max Pool
8. שכבת Fully Connected בגודל 256
9. שכבת פלט בגודל 3



מודל זה הגיע לדיוק של 76.69%, ב epoch 20, ובו גם ערך ה loss היה הנמוך ביותר , אך עדיין גבוהה יחסית לניסיונות האחרים 0.3773. ע"י שינוי של פונקציית האקטיבציה והורדת שכבה , התוצאות של המודל המקורי, הורעו ב 20%. עם זמן הרצה של 50 שניות לepoch.

<u>ניסיון 4:</u>

זהה לניסיון 3, אך עם שינוי פונקציית האקטיבציה ל LReLU עם $\alpha = 0.1$.



מודל זה הגיע לדיוק של 96%, ב epoch 32, ובו גם ערך ה loss היה הנמוך ביותר (0.2133). ע"י שינוי פרמטר α . שינוי זה הגדיל את שיפוע פונקציית האקטיבציה בחלק השלילי, ולכן גרם

לשגיאה של נוירונים שהוציאו ערכים שליליים, להתבטא בצורה טובה יותר. התוצאות של המודל המקורי, השתפרו ב 3%. עם זמן הרצה של כ 50 שניות לepoch.

לאחר בניית המודלים רציתי להעלות תמונה של על שעועית שאינה בדאטאסט ולראות את התיוג שהיא מקבלת במודלים הטובים ביותר מכל סוג. בחרתי במודל ששיפרתי שהתקבל מהניסיון הרביעי . במודל לקחתי את הepoch שקיבל את התוצאות הטובות ביותר ובו הרצתי את התמונה החדשה . השתמשתי בקוד שלמדנו בכיתה לצורך כך. [4]

כאשר העליתי את התמונה החיזוי שגוי לעלה הוא Angular leaf spot. המודל היה ברמת ביטחון של 60% על התיוג של התמונה הזו.

לאחר מכן , הכנסתי תמונה של עלה מלפפון אשר חולה ב Angular leaf spot . סימני המחלה דומים בין העלים השונים ולכן מעניין היה לראות אם המודל שאומן על עלה שעועית יכול לזהות את המחלה הזו גם בעלה מלפפון. [5]

המודל זיהה שמדובר בעלה חולה ב Angular leaf spot ברמת דיוק של 78.8% .

דיון ומסקנות

בעבודה זו לקחתי שני סוגי מודלים של למידה עמוקה :

1. מודל resnet , שהוא מודל שאומן על מיליוני תמונות ונבנה על ידי מומחים רבים בתחום . ישנם כמה סוגים של המודל הזה . אני השתמשתי בשניים מתוכם : resnet 18 ו resnet34 . ההבדל ביניהם הוא מספר השכבות ברשת. רציתי לראות האם הקטנה של התמונה בכל מודל תוביל לדיוק גבוה מספיק , loss נמוך מספיק ובזמן קצר יותר. במודל resnet18 הקטנת התמונה ל 128 הובילה לדיוק גבוה של 94% ובזמן קצר מאוד של 11 שניות לכל epoch. אחרי שהסרתי את הקטנת התמונה באותו המודל קיבלתי דיוק גבוה יותר של 98.4% אך בתמורה לזמן ריצה ארוך יותר של 1:05 דקות לכל epoch . ניתן להבין מכך שהקטנת התמונה פוגעת בדיוק, ולכן אם התמונה לא תוקטן, נקבל דיוק גבוה יותר אך יש trade off של עם זמן אימון ארוך יותר.

כאשר השתמשתי ב resnet34 ללא הקטנת התמונה קיבלתי דיוק של 100% בזמן ריצה של 1:59 דקות לכל epoch . ניתן להבין מכך שהוספה של שכבות יכולה לשפר את הלמידה של המודל , ומאפשרת להגיע אף לדיוק מושלם (עבור סט ה validate) אך במחיר זמן אימון ארוך.

בקוד השתמשתי באוגמנטציה על התמונות, והדבר שיפר את האימון, שכן אוגמנטציה משנה את התמונה בצורה לא מהותית, וכך מאפשרת למודל לראות דוגמאות יותר מגוונות.

2. מודל פשוט יותר של keras . לקחתי מודל שמצאתי ושיפרתי אותו. הורדתי את כמות ה epochים ל 50 כדי למנוע overfitting וכדי לחסוך זמן, והתחלתי בהוספה של מידע לתמונה על ידי הגדלתה (לעומת המודל המקורי). הגדלה לגודל המלא שלה הביא לקריסה של הריצה , לכן הגדלתי את התמונה ל 256 . בגודל הזה קיבלתי דיוק טוב יותר ב 3% . לאחר מכן שיניתי את פונקציית האקטיבציה מ ReLU ל LReLU כדי שגם נוירונים שמוציאים ערכים שליליים יוכלו לתרום לחישוב השגיאה ולעדכון המשקלים של הרשת, והורדתי שכבת ה Fully Connected בגודל 128 כדי לחסוך קצת בזמן ריצה. ב LReLU התחלתי מ$\alpha = 0.05$. הניסיון הזה הביא לתוצאות נמוכות מאוד 76.69% דיוק ולכן ניסיתי להגדיל את α ל 0.1 . בניסיון זה קיבלתי דיוק

של 96% וloss נמוך של 0.2133, שהן תוצאות טובות יותר מהמודל המקורי ובגלל ערך ה loss היחסית נמוך, מצביע על כך שהמודל כנראה ידע להכליל בצורה טובה גם לתמונות חדשות.

מניסיונות אלו ניתן לראות שאפילו בעזרת מודל פשוט יחסית ניתן להגיע לדיוק גבוה. הקטנת התמונה אמנם חוסכת בזמן הריצה, אבל גורמת לאיבוד פרטים ולהפחתת הדיוק, ובנוסף גם פונקציית האקטיבציה בהחלט יכולה להשפיע על התוצאות.

כאשר הכנסתי תמונה של עלה שעועית בריא לתוך הניסיון הרביעי של המודל קיבלתי תגית שגויה שמזהה אותו כחולהAngular leaf spot . וכאשר הכנסתי תמונה של עלה מלפפון החולה ב Angular leaf spot המודל זיהה שזהו עלה החולה במחלה זו . ניתן לראות שהמודל אינו צודק בכל החיזויים שלו אך הוא כן יכול לשמש לזיהוי של אותם גורמי מחלה אשר נלמדו אלין לעלים אחרים. בכפוף שהמחלה מבטאת בצורה דומה בשני העלים .

אם היה ברשותי יותר זמן, הייתי מנסה לאמן את המודל עם ערכים שונים של α בפונקציות האקטיבציה ואף מנסה להשתמש בפונקציית האקטיבציה PReLU שבה השיפוע של פונקציית האקטיבציה בחלקה השלילי נלמד ביחד עם כל שאר המשקלים במודל. בנוסף הייתי מאמנת את המודל על מספר רב יותר של תמונות גם של עלי שעועית וגם של אותן מחלות בלעים אחרים.

ביבליוגרפיה

6–1

1. Almeida CP de, de Carvalho Paulino JF, Bonfante GFJ, et al. Angular Leaf Spot Resistance Loci Associated With Different Plant Growth Stages in Common Bean. *Frontiers in Plant Science*. 2021;12:650. doi:10.3389/FPLS.2021.647043/BIBTEX

2. Knowledge Bank | Bean rust management. Accessed February 20, 2022. https://www.plantwise.org/KnowledgeBank/factsheetforfarmers/20127802224

3. Leaf Bean dataset. Accessed February 20, 2022. https://www.tensorflow.org/datasets/catalog/beans

4. Young bean plant. Accessed February 20, 2022. https://www.dreamstime.com/royalty-free-stock-photos-young-bean-plant-image2289508

5. Cucumber (Cucumis sativus)-Angular Leaf Spot. Accessed February 20, 2022. https://pnwhandbooks.org/plantdisease/host-disease/cucumber-cucumis-sativus-angular-leaf-spot

6. Bean Disease Classification using TensorFlow Convolutional Neural Networks (CNN) | by Ashraf Dasa | Medium. Accessed February 20, 2022. https://medium.com/@ashraf.dasa/bean-disease-classification-using-tensorflow-convolutional-neural-network-cnn-2079dffe87ce#id_token=eyJhbGciOiJSUzI1NiIsImtpZCI6ImFjYjZiZTUxZWZlYTZhNDE5ZWM5MzI1ZmVhYTFlYzQ2NjBmNWIzN2MiLCJ0eXAiOiJKV1QifQ.eyJpc3MiOiJodHRwczovL2FjY291bnRzLmdvb2dsZS5jb20iLCJuYmYiOjE2NDUzNjMzMDQsImF1ZCI6IjIxNjI5NjAzNTgzNC1rMWs2cWUwNjBzMnRwMmEyamFtNGxqZGNtczAwc3R0Zy5hcHBzLmdvb2d

sZXVzZXJjb250ZW50LmNvbSIsInN1YiI6IjEwMDM3NzI3NzQ5NTQzMDI0NTM2OSIsImVt
YWlsIjoiYWtlcm1hbjE5OTdAZ21haWwuY29tIiwiZW1haWxfdmVyaWZpZWQiOnRydW
UsImF6cCI6IjIxNjI5NjAzNTgzNC1rMWs2cWUwNjBzMnRwMmEyamFtNGxqZGNtczAwc
3R0Zy5hcHBzLmdvb2dsZXVzZXJjb250ZW50LmNvbSIsIm5hbWUiOiJBdml0YWwgQWtlc
m1hbiIsInBpY3R1cmUiOiJodHRwczovL2xoMy5nb29nbGV1c2VyY29udGVudC5jb20vYS
9BQVRYQUp5aDBkRG9fNUpUZGxoMzNla090cjl0ZHBocEIwUk1vRm14bzBjSz1zOTYtYy
IsImdpdmVuX25hbWUiOiJBdml0YWwiLCJmYW1pbHlfbmFtZSI6IkFrZXJtYW4iLCJpYXQi
OjE2NDUzNjM2MDQsImV4cCI6MTY0NTM2NzIwNCwianRpIjoiZWNhZjlhMDFkYTJjNm
Q2Yzg1ZGMwZDQxMjBkZWFhMGYwNTg0YWJjOSJ9.V0BZpPhYtg8jNEctLHqiOBAMpB
1En7FYVeKYxOwxjPXzTYR5DFElfcZsCZt96AorpdA6bk5CQWGc0Wu2pPo_chZ432TBc9k
-
Bl7g451LfYhK149XowDO7dTDoB2dSew9cB0_aar3wmNjeTKEXtb1Pfi4fqP65w4ECVZ0F
duL5bdfO6t7nOGXuCsmMLankPsCO_VnNUwJdGKru9EAjwge6LhVqtycaBs-
kVr3RqD_v9UN5Ue0_Nvp4bl0T_EczkcYs-
6YCxzAO70ZgHZpzQd2BNiPje_wSZGwHAhm1bUnCGHrnVeap78ApmoDvcHwRms-
CnfeQK9fwI7sCHhxr5up5w