<u>פרויקט: זיהוי פונטים מתמונות</u>

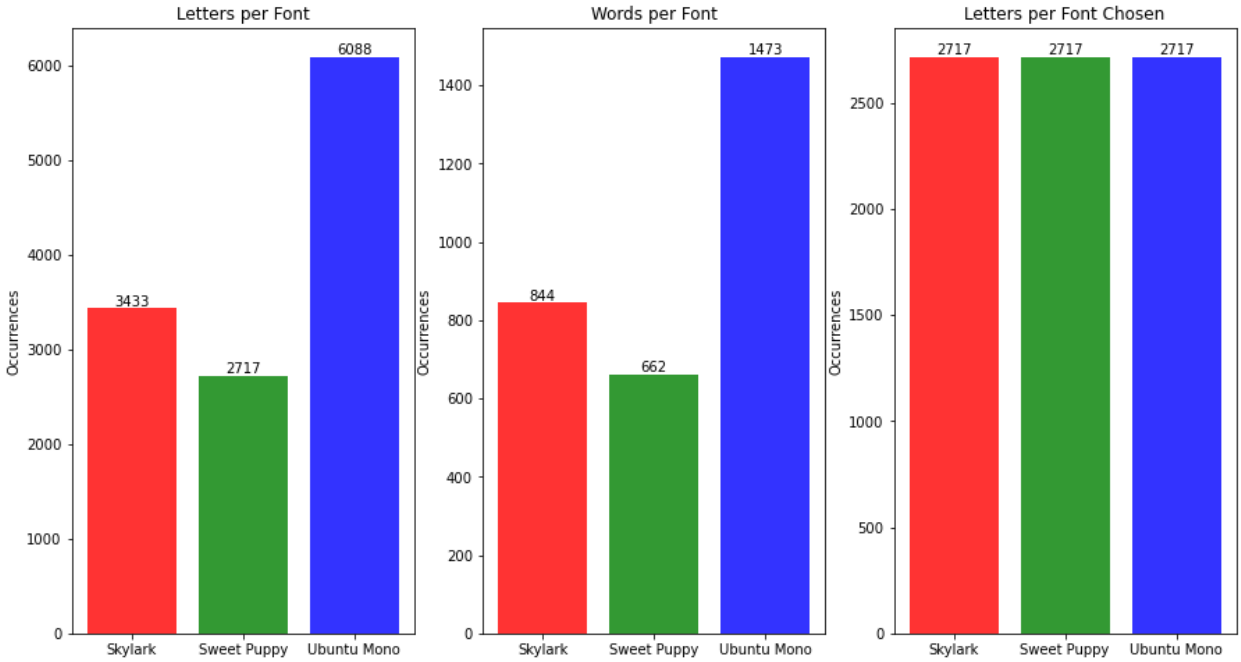
מגיש: רז רמון 315474197

דוח פרויקט:

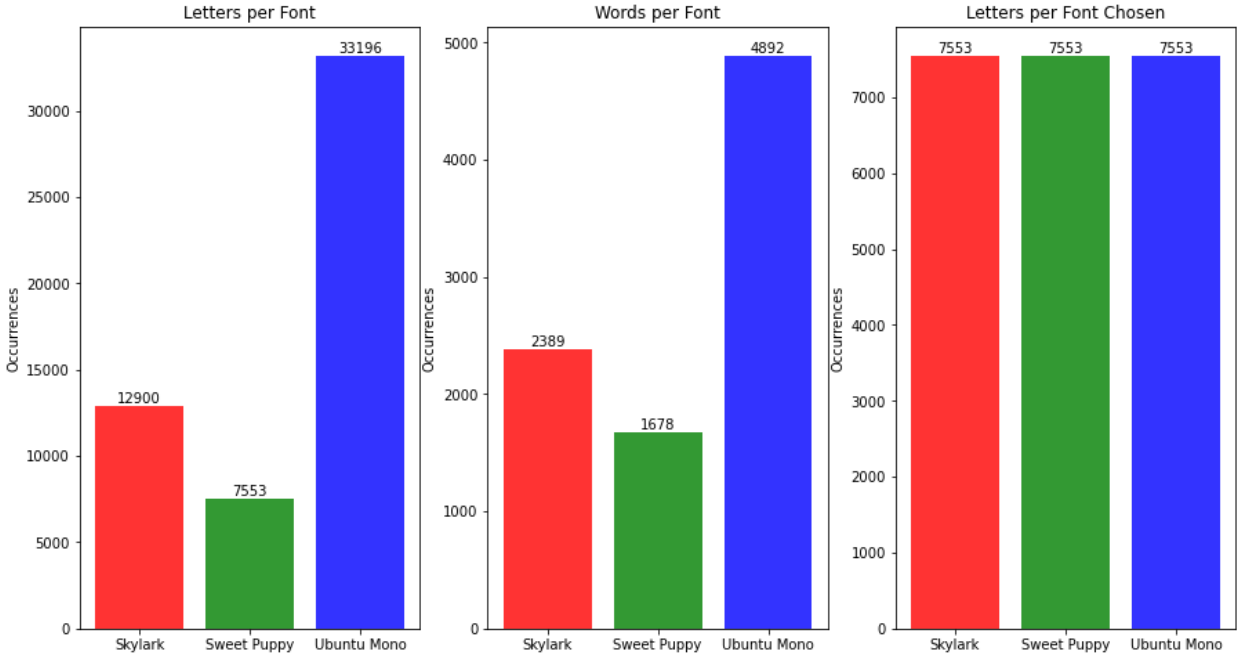
מהלך העבודה

<u>חלק 1 – בדיקת הdataset:</u>

דבר ראשון שעשיתי זאת בדיקה של התמונות – כמה אותיות יש לנו מכל פונט, כמה מילים.
לאחר מכן, על מנת ליצור אחידות ולא bias כלפי פונט מסויים, לקחתי את המינימום של האותיות מבין כל הפונטים – שכן אני מאמן על אותיות, ולא על מילים, אך היה חשוב לראות האם יש קשר בין אורכי המילים לבין הפונטים. הבחירה באותיות נעשתה אחרי ביצוע random על האותיות על מנת ליצור גנרליזציה בין כל המילים וכל הצורות של האותיות.



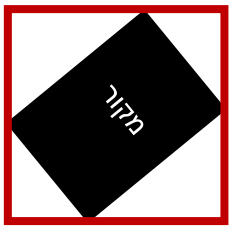
לאחר כמות רבה של ניסיונות על רשתות, ניסיונות להוסיף ולהחסיר שינויים על התמונות – להוסיף רעשים וכו', החלטתי להשתמש בספרייה SynthText על מנת ליצור dataset נוסף לכל סוג פונט בנפרד. הדבר לקח מעל יום, כי הקוד שבGitHub לא עובד, והיה צריך לעשות שינויים בקוד + התאמות של ספריות על מנת שייווצרו תמונות שבהן הbounding boxes יהיו במקומות הנכונים (השגיאה שלקחה הכי הרבה זמן) וגם שיהיה לכל מילה פונט.
כפי שנראה למטה, לאחר הרבה עבודה הצלחתי ליצור עוד תמונות. ללא סיבה שמצאתי, נוצרו הרבה יותר אותיות של הפונט Ubuntu Mono, אך שוב, גם פה נבחרה הכמות המינימלית של האותיות מבין כל הפונטים עבור כולם, שזה עדיין מגן מbias. גם פה נעשה הrandom.



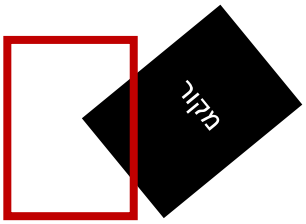
כלומר כמעט שילשתי את כמות האותיות שהמודל מתאמן עליהן, מה שגרם לשיפור ביצועים.

חלק 2 - טיפול בתמונות:

בהינתן הdataset, הייתי צריך איכשהו להוציא את האותיות – על מנת שאוכל ללמד רשת עליהן.
בהתחלה ניסיתי עם חתיכה של כל אות על פי הmax והmin של כל bounding box, אך הדבר יצר בעיה שחלק מהאות אחרת נכנסה לאות הנוכחית והאותיות לא היו מדויקות, אם האות הייתה באלכסון וכתוצאה מכך גם הbounding box שלה, הדבר יצר בעיה שהתמונה שנחתכה הכילה הרבה יותר מאשר רק האות, לכן החלטתי להשתמש בgetPerspectiveTransform ובwarpPerspective על מנת לקחת את הbounding box המקורי ולעשות הטלה על מנת לקבל את התמונה כפי שרציתי.



זה מה שקרה בניסיון המקורי שלי, כאשר באדום זאת האות שאכן נלקחה.



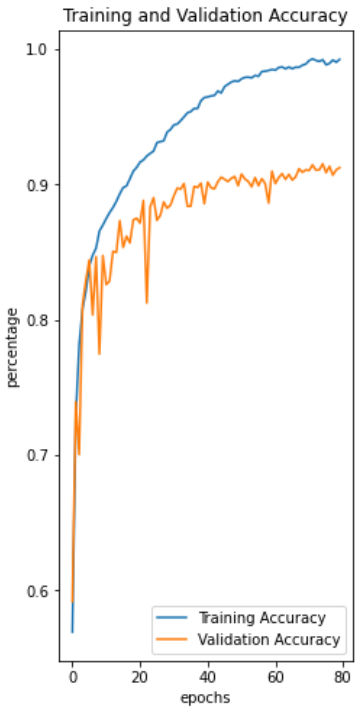
השימוש בgetPerspectiveTransform הטיל את הbounding box כך שתמונה תהיה על ציר הx, וכך תהיה לנו תמונה מיושרת. הwrapPerspective הוא בשביל להיות בדיוק בגודל הרצוי.
התמונות נשמרו ב2 תיקיות שונות, אחת לאימון ואחד לוולידזציה, כאשר כל פונט נשמר בתיקייה נפרדת.

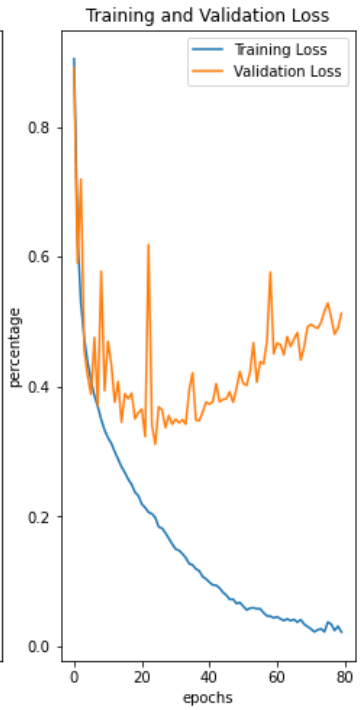
חלק 3 – בניית רשת ראשונה:

החלטתי עוד מההתחלה לעשות בדיקה של זיהוי פונט לאות יחידה לעומת זיהוי פונט למילה – כלומר לסכום את הpredictions על פני כל האותיות במילה ולקחת את המקסימום, ולשים את הפונט הזה לכל האותיות הללו, וכך גם אם אחת מהן לא צלחה, האחרות יתקנו אותה.
לבסוף החלטתי שהמודל שבחרתי יוציא אך ורק את הזיהוי למילה.

כל התוצאות הן עם הוספת עוד dataset, שכן ללא התוצאות לא היו מספיק טובות. כל התוצאות של המילה לעומת האות הן כתוצאה מהרצה על סט הוולידציה שניתן.

הרשת הראשונה שניסיתי:
בהתחלה ניסיתי להבין איך ומה אני רושם, איזו רשת אני מחפש. ממה שלמדנו ומחיפושים התחלתי עם רשת בסיסית, בהשראה של רשת מkaggle, שנועדה לעשות font classification על הdataset notMNIST, שזהו גם dataset של אותיות ופונטים. הרשת שלי מקבלת תמונות אפורות, בהתחלה עושה להן נרמול ע"י rescaling(1./255), לאחר מכן 3 שכבות של conv2D עם maxPooling, ולבסוף flatten ו2 dense לסיום ולמספר הclasses שיש לי וזאת הרשת הראשונה שהרצתי.






```
Layer (type)                 Output Shape              Param #
=================================================================
rescaling (Rescaling)        (None, 256, 256, 1)       0
_________________________________________________________________
conv2d (Conv2D)              (None, 254, 254, 64)      640
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 127, 127, 64)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 125, 125, 128)     73856
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 62, 62, 128)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 60, 60, 256)       295168
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 30, 30, 256)       0
_________________________________________________________________
flatten (Flatten)            (None, 230400)            0
_________________________________________________________________
dropout (Dropout)            (None, 230400)            0
_________________________________________________________________
dense (Dense)                (None, 512)               117965312
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 1539
=================================================================
Total params: 118,336,515
Trainable params: 118,336,515
Non-trainable params: 0
_________________________________________________________________
```

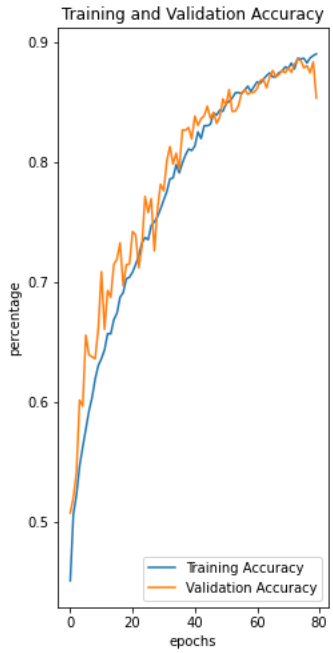
התוצאות על פי – אות/מילה(שמאל מילה, ימין אות):

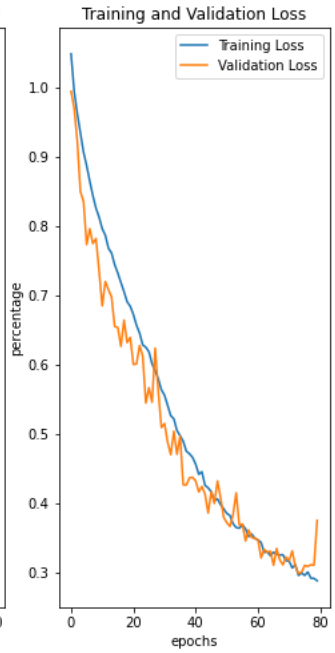
	Precision	recall	f1-score	support
Skylark	0.88/0.88	0.98/0.98	0.93/0.93	501/1915
Sweet Puppy	0.96/0.96	0.97/0.97	0.96/0.96	373/1373
Ubuntu Mono	0.99/0.99	0.93/0.94	0.96/0.96	1133/4910
accuracy			0.95/0.96	2007/8198
macro avg	0.94/0.94	0.96/0.96	0.95/0.95	2007/8198
weighted avg	0.96/0.96	0.95/0.96	0.95/0.96	2007/8198

שאלו תוצאות לא רעות בכלל להרצה הראשונה.
כפי שנראה, הייתי צריך לעשות פחות epochs, על מנת להפסיק את הoverfitting.
התחלתי עם תמונות בגודל 256, שזאת כנראה הבעיה שלי בהמשך.

<u>ניסיונות שיפור של הרשת:</u>
ניסיון 1:
לאחר מכן, החלטתי לנסות לעשות פעולות על הdataset לפני שהוא נכנס לרשת, כלומר לעשות preprocessing כלשהו. לאחר חיפוש מצאתי את הפוסט https://towardsdatascience.com/how-to-reduce-training-time-

for-a-deep-learning-model-using-tf-data-43e1989d2961 שבו מסבירים איך לעשות זאת. הפעולה הזאת נכנסה לכל הניסיונות של הרשת הזאת, כולל ניסיונות גם על הvgg16.






```
Layer (type)                 Output Shape              Param #
=================================================================
sequential (Sequential)      (None, 256, 256, 1)       0
_________________________________________________________________
rescaling (Rescaling)        (None, 256, 256, 1)       0
_________________________________________________________________
conv2d (Conv2D)              (None, 254, 254, 64)      640
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 127, 127, 64)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 125, 125, 128)     73856
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 62, 62, 128)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 60, 60, 256)       295168
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 30, 30, 256)       0
_________________________________________________________________
flatten (Flatten)            (None, 230400)            0
_________________________________________________________________
dropout (Dropout)            (None, 230400)            0
_________________________________________________________________
dense (Dense)                (None, 512)               117965312
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 1539
=================================================================
Total params: 118,336,515
Trainable params: 118,336,515
Non-trainable params: 0
_________________________________________________________________
```

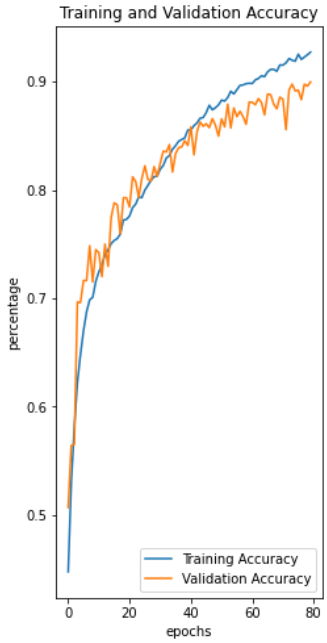
התוצאות על פי – אות/מילה(שמאל מילה, ימין אות):

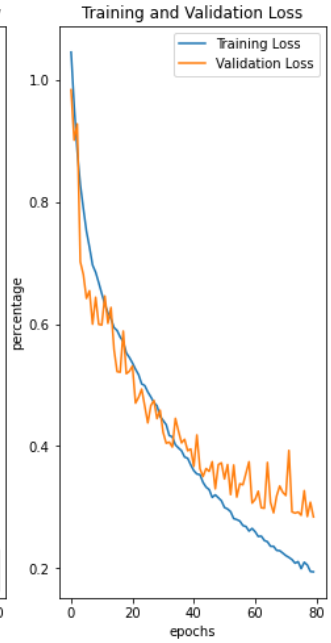
	Precision	recall	f1-score	support
Skylark	0.86/0.86	0.95/0.95	0.91/0.90	501/1915
Sweet Puppy	0.81/0.79	0.98/0.98	0.89/0.88	373/1373
Ubuntu Mono	0.98/0.98	0.87/0.88	0.92/0.93	1133/4910
accuracy			0.91/0.91	2007/8198
macro avg	0.89/0.88	0.93/0.94	0.91/0.90	2007/8198
weighted avg	0.92/0.92	0.91/0.91	0.91/0.91	2007/8198

כלומר חלק מהpreprocessing הרסו, לכן אחרי כמה ניסיונות מגיע ניסיון 2.

ניסיון 2:

כמו כן, על מנת ליעל את הרשת, על פי הפוסט הזה https://datascience.stackexchange.com/questions/22760/number-and-size-of-dense-layers-in-a-cnn החלטתי להוסיף עוד שכבת dense בסוף, ולהוריד ולשנות חלק מהpreprocessing layers.





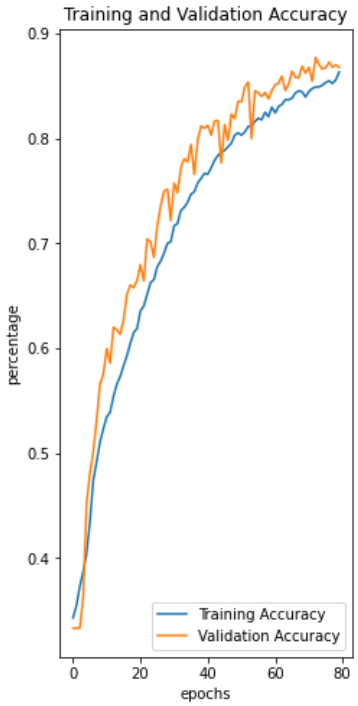

```
Layer (type)                 Output Shape              Param #
=================================================================
sequential_2 (Sequential)    (None, 256, 256, 1)       0
_________________________________________________________________
rescaling_1 (Rescaling)      (None, 256, 256, 1)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 254, 254, 64)      640
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 127, 127, 64)      0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 125, 125, 128)     73856
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 62, 62, 128)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 60, 60, 256)       295168
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 30, 30, 256)       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 230400)            0
_________________________________________________________________
dropout_1 (Dropout)          (None, 230400)            0
_________________________________________________________________
dense_3 (Dense)              (None, 256)               58982656
_________________________________________________________________
dense_4 (Dense)              (None, 512)               131584
_________________________________________________________________
dense_5 (Dense)              (None, 3)                 1539
=================================================================
Total params: 59,485,443
Trainable params: 59,485,443
Non-trainable params: 0
_________________________________________________________________
```

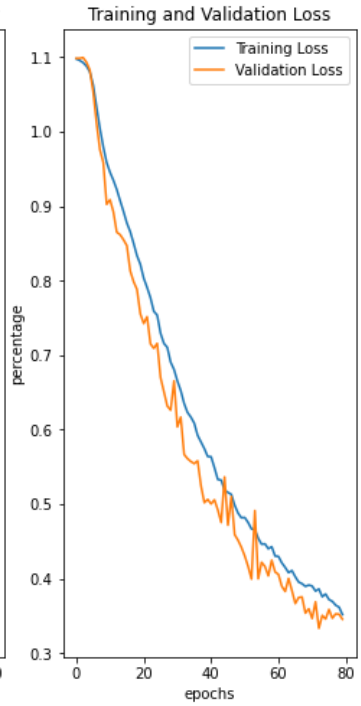
התוצאות על פי – אות/מילה(שמאל מילה, ימין אות):

	Precision	recall	f1-score	support
Skylark	0.83/0.82	0.97/0.97	0.89/0.89	501/1915
Sweet Puppy	0.91/0.89	0.94/0.95	0.93/0.92	373/1373
Ubuntu Mono	0.99/0.99	0.90/0.90	0.94/0.94	1133/4910
accuracy			0.92/0.92	2007/8198
macro avg	0.91/0.90	0.94/0.94	0.92/0.92	2007/8198
weighted avg	0.93/0.93	0.92/0.92	0.92/0.92	2007/8198

ניסיון 3:

הכפלת כל שכבת קונבולוציה וכל שכבת pooling, כאשר בין כל שינוי גודל של הקונבולוציה, ביצעתי dropout.






```
Layer (type)                    Output Shape              Param #
=================================================================
sequential (Sequential)         (None, 256, 256, 1)       0
_________________________________________________________________
rescaling_4 (Rescaling)         (None, 256, 256, 1)       0
_________________________________________________________________
conv2d_24 (Conv2D)              (None, 254, 254, 64)      640
_________________________________________________________________
max_pooling2d_24 (MaxPooling    (None, 127, 127, 64)      0
_________________________________________________________________
conv2d_25 (Conv2D)              (None, 125, 125, 64)      36928
_________________________________________________________________
max_pooling2d_25 (MaxPooling    (None, 62, 62, 64)        0
_________________________________________________________________
dropout_24 (Dropout)            (None, 62, 62, 64)        0
_________________________________________________________________
conv2d_26 (Conv2D)              (None, 60, 60, 128)       73856
_________________________________________________________________
max_pooling2d_26 (MaxPooling    (None, 30, 30, 128)       0
_________________________________________________________________
conv2d_27 (Conv2D)              (None, 28, 28, 128)       147584
_________________________________________________________________
max_pooling2d_27 (MaxPooling    (None, 14, 14, 128)       0
_________________________________________________________________
dropout_25 (Dropout)            (None, 14, 14, 128)       0
_________________________________________________________________
conv2d_28 (Conv2D)              (None, 12, 12, 256)       295168
_________________________________________________________________
max_pooling2d_28 (MaxPooling    (None, 6, 6, 256)         0
_________________________________________________________________
conv2d_29 (Conv2D)              (None, 4, 4, 256)         590080
_________________________________________________________________
max_pooling2d_29 (MaxPooling    (None, 2, 2, 256)         0
_________________________________________________________________
flatten_4 (Flatten)             (None, 1024)              0
_________________________________________________________________
dropout_26 (Dropout)            (None, 1024)              0
_________________________________________________________________
dense_12 (Dense)                (None, 256)               262400
_________________________________________________________________
dense_13 (Dense)                (None, 512)               131584
_________________________________________________________________
dense_14 (Dense)                (None, 3)                 1539
=================================================================
Total params: 1,539,779
Trainable params: 1,539,779
Non-trainable params: 0
```

התוצאות על פי – אות/מילה(שמאל מילה, ימין אות):

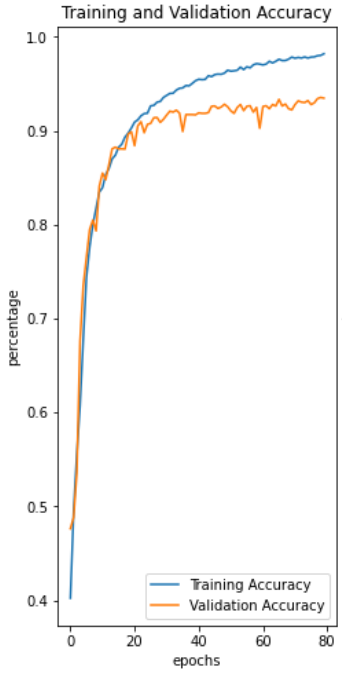
	Precision	recall	f1-score	support
Skylark	0.76/0.75	0.99/0.98	0.86/0.85	501/1915
Sweet Puppy	0.88/0.95	0.98/0.98	0.93/0.92	373/1373
Ubuntu Mono	0.99/0.99	0.83/0.84	0.91/0.91	1133/4910
accuracy			0.90/0.90	2007/8198
macro avg	0.88/0.87	0.93/0.93	0.90/0.89	2007/8198
weighted avg	0.92/0.92	0.90/0.90	0.90/0.90	2007/8198

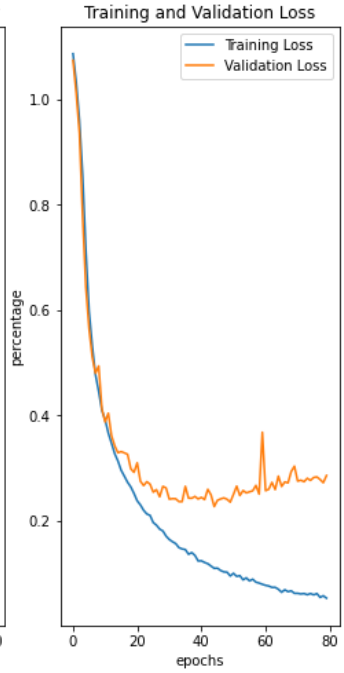
כפי שנראה, כנראה ביצעתי יותר מדיי dropout, לכן התוצאות השתבשו ולא הצליחו.

ניסיון 4:

הקטנה של התמונות לגודל 100, הוספה של עוד שכבה של cov2d בגודל 512 ואחריה maxPooling, כמו כן, גם פה יש שינוי של הdropouts כך שבין כמעט כל השכבות קיימת שכבה כזאת.

הרשת נתנה תוצאות יפות, גם ללא data augmentation.





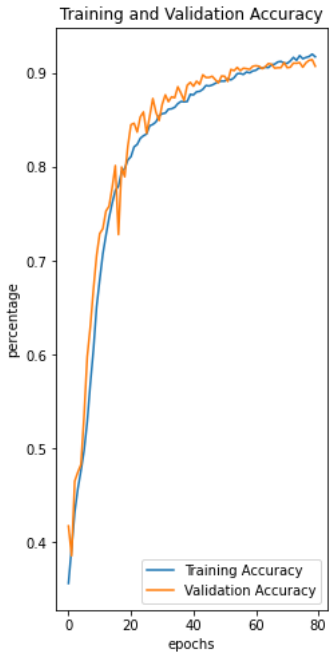

```
Layer (type)                 Output Shape              Param #
=================================================================
rescaling (Rescaling)        (None, 100, 100, 1)       0
_________________________________________________________________
conv2d (Conv2D)              (None, 98, 98, 64)        640
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 49, 49, 64)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 47, 47, 128)       73856
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 23, 23, 128)       0
_________________________________________________________________
dropout (Dropout)            (None, 23, 23, 128)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 21, 21, 256)       295168
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 10, 10, 256)       0
_________________________________________________________________
dropout_1 (Dropout)          (None, 10, 10, 256)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 512)         1180160
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 4, 4, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 8192)              0
_________________________________________________________________
dropout_2 (Dropout)          (None, 8192)              0
_________________________________________________________________
dense (Dense)                (None, 512)               4194816
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 1539
=================================================================
Total params: 5,746,179
Trainable params: 5,746,179
Non-trainable params: 0
_________________________________________________________________
```

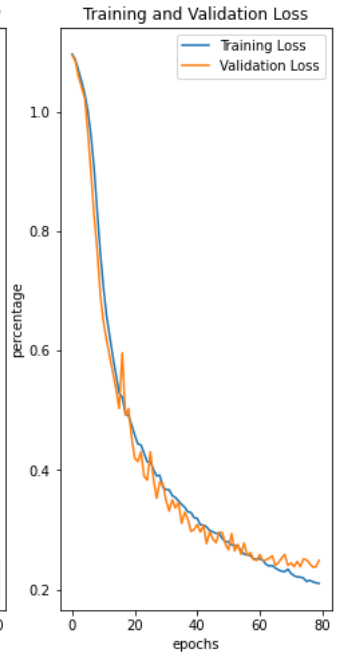
התוצאות על פי – אות/מילה(שמאל מילה, ימין אות):

	Precision	recall	f1-score	support
Skylark	0.94/0.94	0.97/0.97	0.95/0.95	501/1915
Sweet Puppy	0.96/0.96	0.98/0.98	0.97/0.97	373/1373
Ubuntu Mono	0.98/0.99	0.96/0.96	0.97/0.97	1133/4910
accuracy			0.97/0.97	2007/8198
macro avg	0.96/0.96	0.97/0.97	0.97/0.97	2007/8198
weighted avg	0.97/0.97	0.97/0.97	0.97/0.97	2007/8198

ניסיון 5:

הרשת הקודמת עם data augmentation.



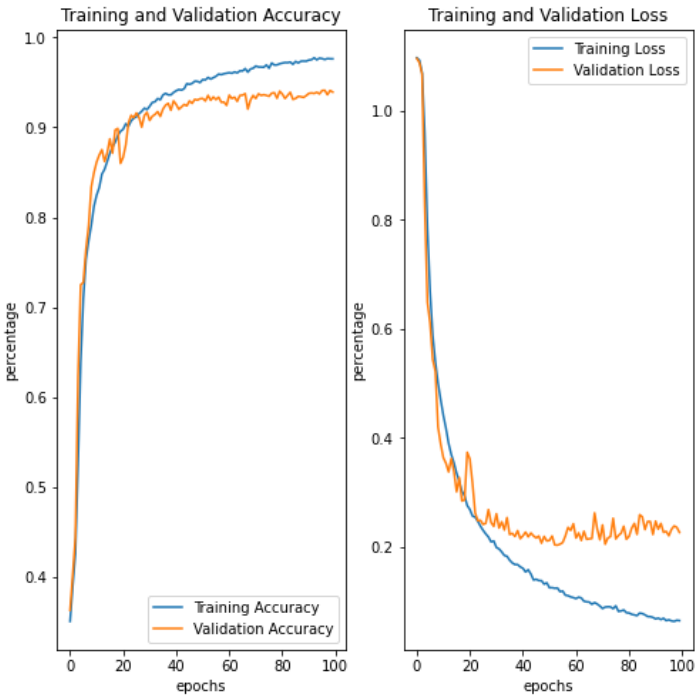



```
Layer (type)                 Output Shape              Param #
=================================================================
sequential (Sequential)      (None, 100, 100, 1)       0
_________________________________________________________________
rescaling (Rescaling)        (None, 100, 100, 1)       0
_________________________________________________________________
conv2d (Conv2D)              (None, 98, 98, 64)        640
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 49, 49, 64)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 47, 47, 128)       73856
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 23, 23, 128)       0
_________________________________________________________________
dropout (Dropout)            (None, 23, 23, 128)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 21, 21, 256)       295168
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 10, 10, 256)       0
_________________________________________________________________
dropout_1 (Dropout)          (None, 10, 10, 256)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 512)         1180160
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 4, 4, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 8192)              0
_________________________________________________________________
dropout_2 (Dropout)          (None, 8192)              0
_________________________________________________________________
dense (Dense)                (None, 512)               4194816
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 1539
=================================================================
Total params: 5,746,179
Trainable params: 5,746,179
Non-trainable params: 0
_________________________________________________________________
```

התוצאות על פי – אות/מילה(שמאל מילה, ימין אות):

	Precision	recall	f1-score	support
Skylark	0.94/0.93	0.98/0.98	0.96/0.95	501/1915
Sweet Puppy	0.95/0.95	0.98/0.99	0.97/0.97	373/1373
Ubuntu Mono	0.99/0.99	0.96/0.96	0.97/0.98	1133/4910
accuracy			0.97/0.97	2007/8198
macro avg	0.96/0.96	0.97/0.97	0.97/0.97	2007/8198
weighted avg	0.97/0.97	0.97/0.97	0.97/0.97	2007/8198

ניסיון 6 (הרשת שנבחרה!): הוספה של שכבת conv2d ושכבת maxpooling באמצע, עם תמונות בגודל 100.




```
Layer (type)                 Output Shape              Param #
=================================================================
sequential_3 (Sequential)    (None, 100, 100, 1)       0
_________________________________________________________________
rescaling_2 (Rescaling)      (None, 100, 100, 1)       0
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 98, 98, 64)        640
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 49, 49, 64)        0
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 47, 47, 128)       73856
_________________________________________________________________
max_pooling2d_12 (MaxPooling (None, 23, 23, 128)       0
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 21, 21, 256)       295168
_________________________________________________________________
max_pooling2d_13 (MaxPooling (None, 10, 10, 256)       0
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 8, 8, 256)         590080
_________________________________________________________________
max_pooling2d_14 (MaxPooling (None, 4, 4, 256)         0
_________________________________________________________________
dropout_4 (Dropout)          (None, 4, 4, 256)         0
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 2, 2, 512)         1180160
_________________________________________________________________
max_pooling2d_15 (MaxPooling (None, 1, 1, 512)         0
_________________________________________________________________
flatten_2 (Flatten)          (None, 512)               0
_________________________________________________________________
dropout_5 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_4 (Dense)              (None, 512)               262656
_________________________________________________________________
dense_5 (Dense)              (None, 3)                 1539
=================================================================
Total params: 2,404,099
Trainable params: 2,404,099
Non-trainable params: 0
_________________________________________________________________
```

	Precision	recall	f1-score	support
Skylark	0.95/0.96	0.98/0.98	0.97/0.97	501/1915
Sweet Puppy	0.97/0.97	0.99/0.99	0.98/0.98	373/1373
Ubuntu Mono	0.99/0.99	0.97/0.97	0.98/0.98	1133/4910
accuracy			0.98/0.98	2007/8198
macro avg	0.97/0.97	0.98/0.98	0.98/0.98	2007/8198
weighted avg	0.98/0.98	0.98/0.98	0.98/0.98	2007/8198

זוהי הרשת עם התוצאות הטובות ביותר שיצא לי, לכן בחרתי לקחת אותה.

לסיום על הרשת שלי:

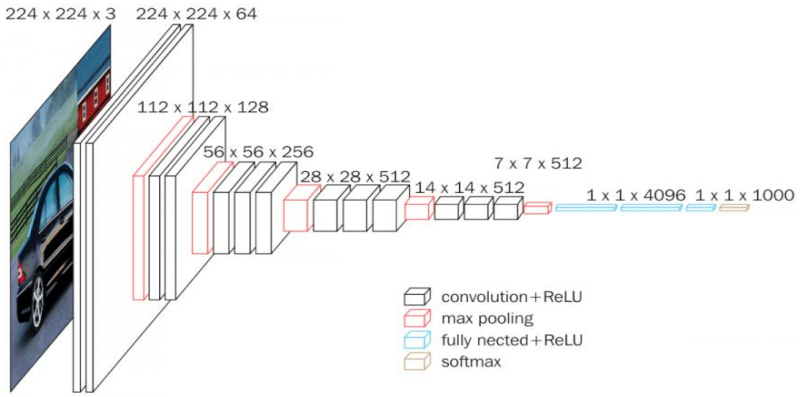
נראה שכשהתחלתי עם תמונות גדולות, הדבר היה בעייתי, שכן כשלרשת היו כל כך הרבה פיקסלים, התוצאות נהיו גרועות וכנראה שהרשת לא הצליחה להתאמן כראוי.
בעת הורדת הרזולוציה של התמונות, הרשת הצליחה להתאמן, יותר מהר ועם תוצאות יותר טובות.
הייתי צריך לחקור על כל הdata augmentation, על מנת לחשוב מה מתאים לי ולעשות הרבה ניסיונות עם פרמטרים שונים, עד שהגעתי לתוצאה הכי טובה.

חלק 4 – עבודה על פי המאמר ותוצאות של VGG16:

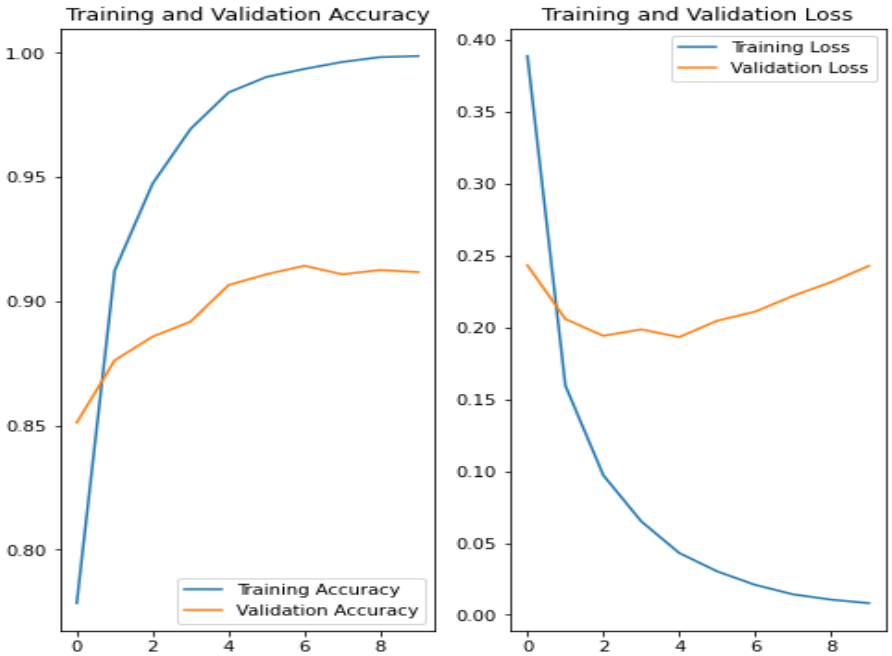
בעקבות המאמר:

"Font Recognition in Natural Images via Transfer Learning"
https://www.icst.pku.edu.cn/zlian/docs/20181024110641005904.pdf

החלטתי לנסות ולהשתמש ברשת קיימת, בטכניקה שנקראת transfer learning, שבה משתמשים ברשת קיימת מאומנת, כלומר שיש לה כבר משקולות לאחר training, וללמד אותה את הdataset הקיים על מנת לזהות את הפונטים. הרשת שאני בחרתי אומנה על imagenet, שזהו dataset ענק עם מעל 14 מיליון תמונות מתויגות ששייכות ל1000 classes, לכן הרשת מאומנת כבר על מנת לבצע classification על תמונות.
הרשת היא VGG16, אשר מתוארת כך:



בהתחלה לקחתי את הרשת לבדה, וניסיתי לאמן ללא המשקולות הקיימות של imagenet, אך הדבר לא הצליח בכלל, ולאחר מכן בחרתי כן להשתמש בimagenet, שהצליח בצורה מעולה על ההתחלה, עוד לפני שניסיתי לשנות משהו. על מנת להתגבר על כך שהרשת מיועדת למספר classes בגודל 1000, הוספתי סיום של 2 שכבות – dense וsoftmax בגודל 3 – שיתאים למספר הfonts\classes שיש לנו.
כפי שנראה, לא הייתי צריך הרבה epochs, שכן כבר החל מ10 התחיל overfitting, לכן בחרתי לעצור ב10.



התוצאות תמיד חושבו גם על פי כל אות בנפרד וגם על פי כל מילה – סכמתי את התוצאות, ושמתי את המקסימום בשביל כל האותיות במילה – כלומר גם אם אות אחת סטתה\לא נבחרה נכון, נתקן בעזרת שאר האותיות במילה.

התוצאות על פי – אות/מילה(שמאל מילה, ימין אות):

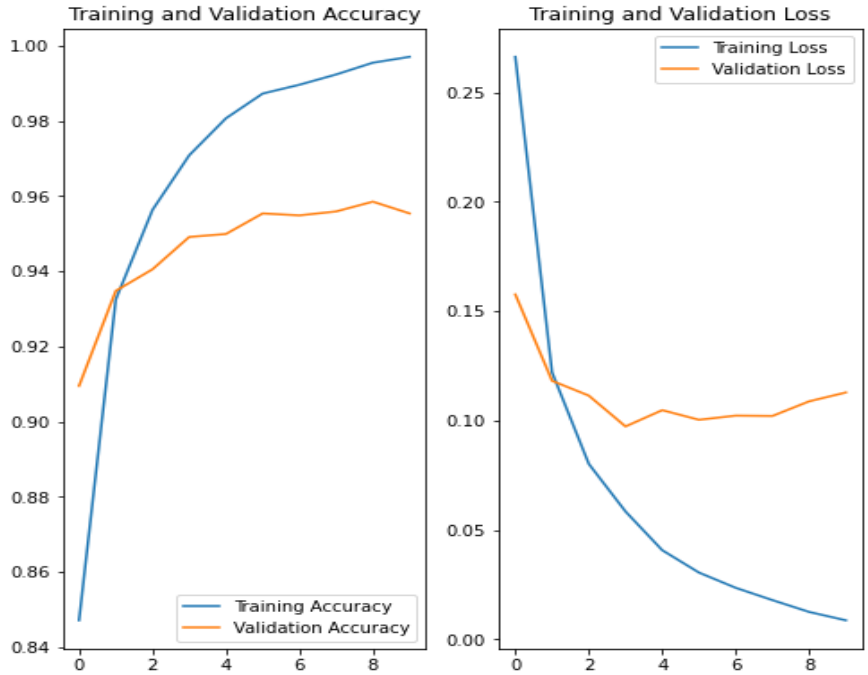
	Precision	recall	f1-score	support
Skylark	0.95/0.95	0.96/0.96	0.96/0.95	501/1915

Sweet Puppy	0.96/0.95	0.97/0.97	0.97/0.96	373/1373
Ubuntu Mono	0.98/0.98	0.97/0.97	0.98/0.98	1133/4910
accuracy			0.97/0.97	2007/8198
macro avg	0.96/0.96	0.97/0.97	0.97/0.96	2007/8198
weighted avg	0.97/0.97	0.97/0.97	0.97/0.97	2007/8198

ניסיונות שיפור של רשת VGG16:

ניסיון ראשון:

הדבר הראשון שרציתי לעשות הוא להוסיף את הdata החדש לשתי הרשתות ולראות מה קורה לתוצאות.
באופן מוזר, דווקא לvgg16 קצת ירד הדיוק.
Vgg16 עם תוספת של data:

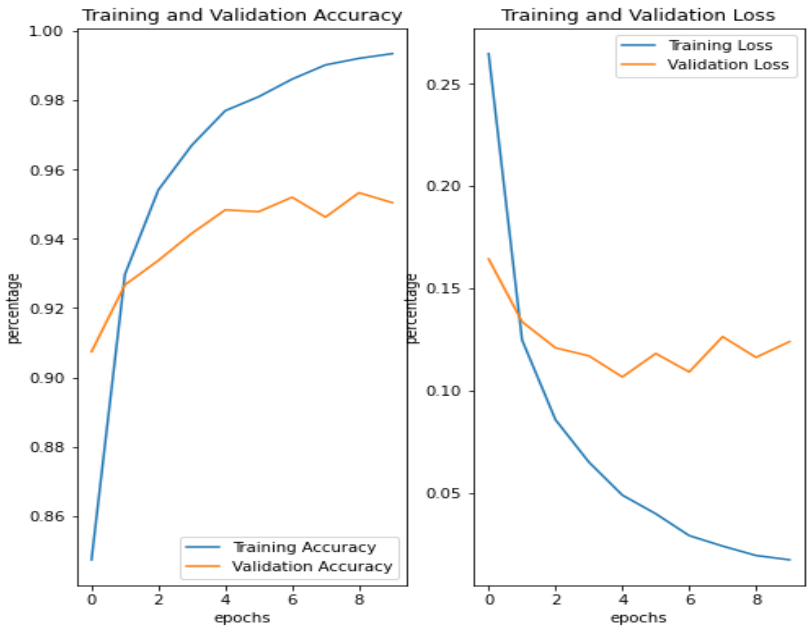


התוצאות על פי – אות/מילה:

	Precision	recall	f1-score	support
Skylark	0.92/0.91	0.97/0.97	0.95/0.94	501/1915
Sweet Puppy	0.95/0.94	0.98/0.97	0.96/0.96	373/1373
Ubuntu Mono	0.99/0.99	0.95/0.95	0.97/0.97	1133/4910
accuracy			0.96/0.96	2007/8198
macro avg	0.95/0.95	0.97/0.97	0.96/0.96	2007/8198
weighted avg	0.96/0.96	0.96/0.96	0.96/0.96	2007/8198

ניסיון שני:

Vgg16 עם תוספת של data ועם layers של salt and pepper ורעש גאוסיאני:

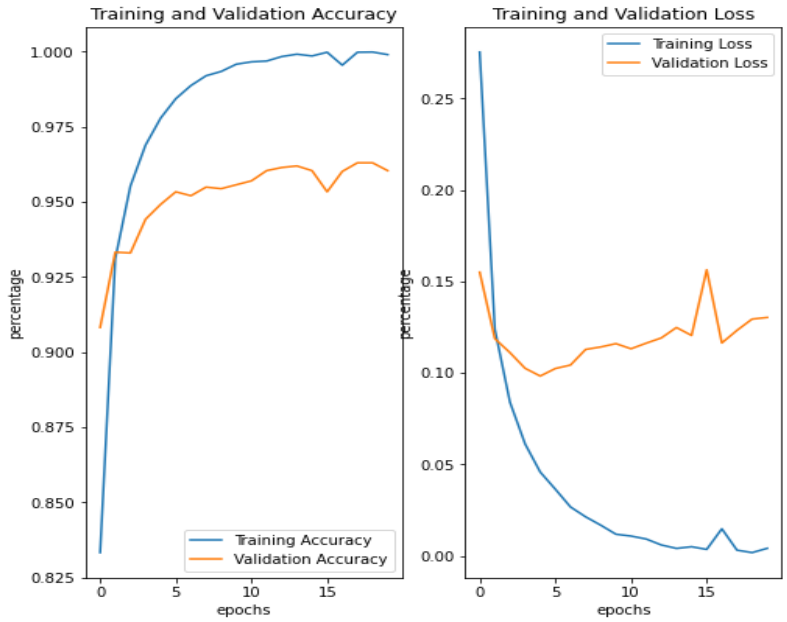


התוצאות על פי – אות/מילה:

	Precision	recall	f1-score	support
Skylark	0.95/0.95	0.97/0.97	0.96/0.96	501/1915
Sweet Puppy	0.95/0.95	0.99/0.98	0.97/0.96	373/1373
Ubuntu Mono	0.99/0.99	0.96/0.97	0.98/0.97	1133/4910
accuracy			0.97/0.97	2007/8198
macro avg	0.96/0.96	0.97/0.97	0.97/0.97	2007/8198
weighted avg	0.97/0.97	0.97/0.97	0.97/0.97	2007/8198

ניסיון 3:

הקטנת גודל התמונות ל150 על 150 במקום 224 על 224.



	Precision	recall	f1-score	support
Skylark	0.94/0.94	0.98/0.99	0.96/0.96	501/1915
Sweet Puppy	0.95/0.94	0.99/0.99	0.97/0.96	373/1373
Ubuntu Mono	0.99/0.99	0.96/0.96	0.98/0.98	1133/4910
accuracy			0.97/0.97	2007/8198
macro avg	0.96/0.96	0.96/0.98	0.97/0.97	2007/8198
weighted avg	0.97/0.97	0.97/0.97	0.97/0.97	2007/8198

לסיום:
על כן ולאחר כל הניסיונות, הבחירה הייתה בין הvgg16 עם התוספות של הdata augmentation לבין הרשת שלי עם התמונות הקטנות והdata augmentation, כלומר ניסיון 6, החלטתי לקחת את הרשת שאני יצרתי עם התוספות, שכן לפי דעתי זה ייתן את הgeneralization הטוב ביותר, בזמן אימון וזמן ריצה הטובים ביותר.

בנימה אישית:
נהניתי מאוד מהעבודה, הרבה זמן רציתי להיכנס לתחום וזה בהחלט הכריח אותי לעשות את הצעד סופסוף, להבין הרבה יותר ולחקור בעצמי, ואפילו לחבר את הGPU שלי על מנת להריץ את המודלים במהירות הטובה ביותר.

נספחים:

תוצאות של הרצת הרשתות עם image_dataset_from_directory כאשר validation_split הוא 0.2:

https://stackoverflow.com/questions/45117295/what-is-the-relation-between-validation-data-and-validation-split-in-keras-fit

על פי התשובה, הבעיה עם הדבר הזה שאין לדעת איך keras ייקח מהדאטה – כלומר יכול להיות שהוא ייקח יותר מפונט אחד מאשר מהשני. כפי שהתוצאות מראות, הדבר כנראה נכון, שכן יש הטיה כלפי הפונט UbuntuMono.

הרצתי רוב הזמן את הרשתות עם זה, ולא הבנתי למה תמיד התוצאות על האחרים פחות טובות, לכן הלכתי לבדוק.
התוצאות על הריצות:

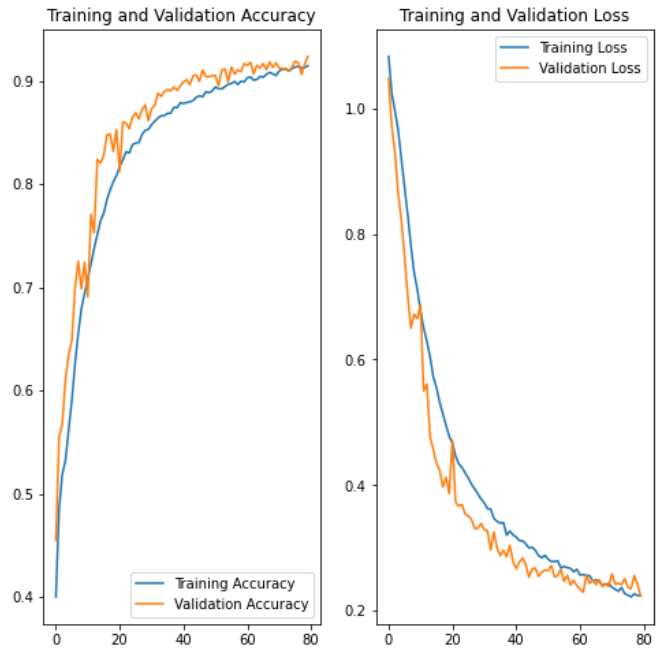
המודל VGG:
התוצאות על פי – אות/מילה:

	Precision	recall	f1-score	support
Skylark	0.96/0.96	0.98/0.98	0.97/0.97	501/1915
Sweet Puppy	0.92/0.91	0.99/0.99	0.95/0.95	373/1373
Ubuntu Mono	0.99/0.99	0.96/0.96	0.97/0.98	1133/4910
accuracy			0.97/0.97	2007/8198
macro avg	0.95/0.95	0.97/0.97	0.96/0.96	2007/8198
weighted avg	0.97/0.97	0.97/0.97	0.97/0.97	2007/8198

המודל שלי:
התוצאות על פי – אות/מילה:

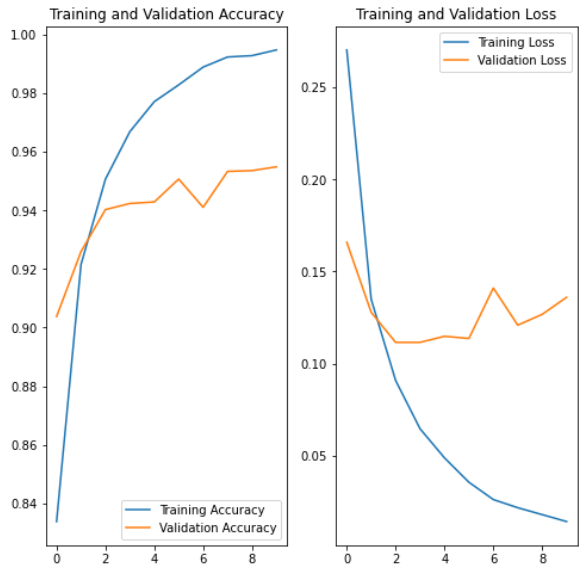
	Precision	recall	f1-score	support
Skylark	0. 93/0.93	0.99/0.99	0.96/0.96	501/1915
Sweet Puppy	0.96/0.95	0.97/0.98	0.97/0.97	373/1373

Ubuntu Mono	0.99/0.99	0.96/0.96	0.98/0.98	1133/4910
accuracy			0.97/0.97	2007/8198
macro avg	0.96/0.96	0.97/0.98	0.97/0.97	2007/8198
weighted avg	0.97/0.97	0.97/0.97	0.97/0.97	2007/8198



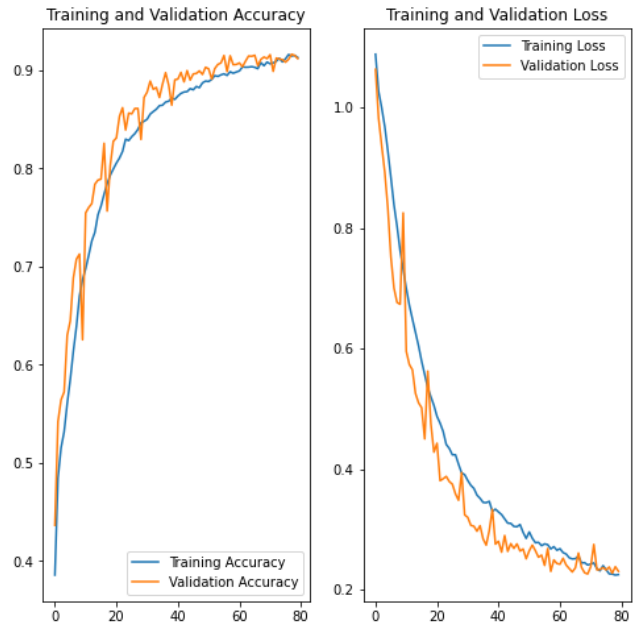
המודל VGG עם data augmentation:
התוצאות על פי – אות/מילה:

	Precision	recall	f1-score	support
Skylark	0. 90/0.89	0.98/0.98	0.94/0.93	501/1915
Sweet Puppy	0.91/0.91	0.99/0.99	0.95/0.95	373/1373
Ubuntu Mono	1.00/1.00	0.93/0.94	0.96/0.97	1133/4910
accuracy			0.95/0.96	2007/8198
macro avg	0.94/0.93	0.97/0.97	0.95/0.95	2007/8198
weighted avg	0.96/0.96	0.95/0.96	0.96/0.96	2007/8198



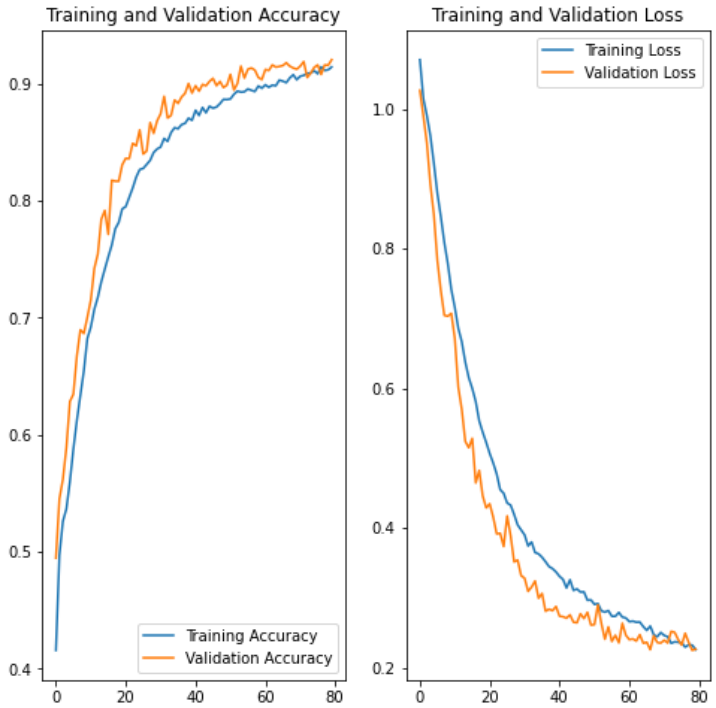
המודל שלי עם רעש גאוסיאני:
התוצאות על פי – אות/מילה:

	Precision	recall	f1-score	support
Skylark	0. 92/0.92	0.98/0.98	0.95/0.95	501/1915
Sweet Puppy	0.95/0.94	0.97/0.97	0.96/0.96	373/1373
Ubuntu Mono	0.99/0.99	0.96/0.96	0.97/0.97	1133/4910
accuracy			0.96/0.96	2007/8198
macro avg	0.95/0.95	0.97/0.97	0.96/0.96	2007/8198
weighted avg	0.97/0.97	0.96/0.96	0.96/0.97	2007/8198



המודל שלי עם רעש גאוסיאני ורעש salt and pepper:
התוצאות על פי – אות/מילה:

	Precision	recall	f1-score	support
Skylark	0. 91/0.91	0.99/0.99	0.95/0.95	501/1915
Sweet Puppy	0.95/0.94	0.97/0.98	0.96/0.96	373/1373
Ubuntu Mono	0.99/0.99	0.95/0.96	0.97/0.97	1133/4910
accuracy			0.96/0.96	2007/8198
macro avg	0.95/0.95	0.97/0.97	0.96/0.96	2007/8198
weighted avg	0.96/0.97	0.96/0.96	0.96/0.96	2007/8198


Training and Validation Accuracy
Training Accuracy
Validation Accuracy
0.9
0.8
0.7
0.6
0.5
0.4
0
20
40
60
80
Training and Validation Loss
Training Loss
Validation Loss
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80