מבוא לראייה ממוחשבת 22928

דו"ח פרויקט גמר

מגיש: נועם שדה

תאריך: 14/02/2023

פרויקט זה עוסק בבעיית סיווג פונטים המוטבעים בתוך תמונה. התבקשנו לייצר מודל אשר ידע לסווג 5 פונטים מסוגים שונים:

Alex Brush Regular

Open Sans Regular

Sansation

Ubuntu Mono

Titillium Web

<u>סיכום תהליך יצירת המודל:</u>

1. הבנת בסיס הנתונים שקיבלנו והצגת פרטים רלוונטים
2. עיבוד מקדים של אוסף התמונות וחלוקת הנתונים לסט של אימון וסט של ולידציה(preprocessing)
3. בניית המודל בעזרת רשת קונבולוציה (convolutional neural network)
4. הערכת התוצאות
5. אופטימיזציה והערכת תוצאות מחודשת
6. שמירת המודל ובדיקה על סט הולידציה

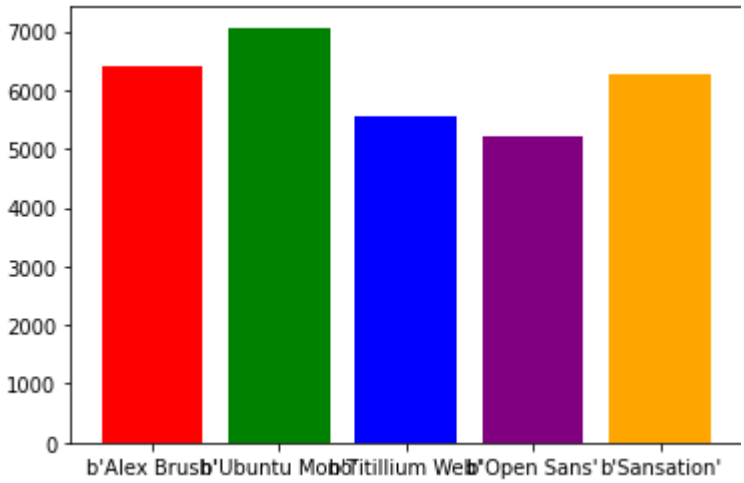
חלק ראשון: הבנת בסיס הנתונים

הקובץ שקיבלנו הינו מסוג HDF5 המכיל:

- 998 תמונות עם אותיות שמוטבעות בהם
- מילים בכל תמונה
- מספר אקראי של אותיות בכל מילה, בסך הכל יש 30520 אותיות
- לכל אות – פונט מתאים. בסה"כ יש 5 פונטים
- תיבה תוחמת של כל מילה
- תיבה תוחמת של כל אות

תמונה לדוגמא:

תחילה בדקתי כמה תמונות של אותיות יש מכל קטגוריה:



יש הכי הרבה אותיות מסוג Ubuntu, אך לא בהרבה ולכן לא איזנתי את סט הנתונים.

חלק שני: עיבוד התמונות וחלוקת הנתונים לסט של אימון ושל ולידציה

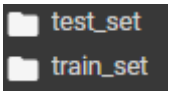
כדי לעבד את התמונות השתמשתי בטכניקת ההומוגרפיה, אשר למדנו בקורס.
משתמשים בטכניקה זו כדי לתקן עיוות פרספקטיבה, ליישר תמונות ועוד. בגלל שהטקסט המוטבע בתמונות הוא בגדלים שונים, זוויות שונות, ובצורות שונות, הומוגורפיה היא אכן פתרון טוב.
לאחר מכן חתכתי את התמונות של לגודל של 64X64 פיקסלים על מנת שיהיה קל יותר לאמן את רשת הקונבולוציה ובצורה יותר מדויקת, וגם לא אאבד מידע רלוונטי.
בנוסף, 10 פיקסלים בכל צד הוקצו כשוליים של התו על מנת להבין בצורה ברורה יותר את התו.

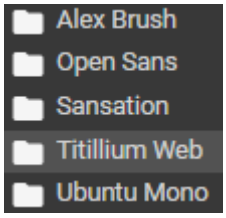
דוגמא חיתוך של תו אחרי שימוש בהומוגרפיה לעומת בלי:

עם הומוגרפייה:  בלי: 

כעט ניתן לראות כל אות באופן ברור.

כדי להכין את סט האימון ואת סט הולידציה יצרתי תיקיות באופן הבא:

- תיקייה של המודל הנקראת model שבה יהיו הקבצים של המודל
- תיקייה של סט אימון ותיקייה של סט ולידציה: 

  - בכל תיקייה כזו: יצרתי תיקייה לכל פונט: 

חילקתי את סט האימון וסט הולידציה לפי **התמונות** ולא לפי אותיות מכיוון שכשנקבל את סט המבחן הוא יהיה מורכב מתמונות שהמודל עוד לא ראה. חילקתי הנתונים כך ש 80% יהיו בסט האימון ו 20% בסט הולידציה כפי שנהוג.
שמרתי את הקבצים כקבצי bmp בתיקיות המתאימות.
לצורך פשטות האות הראשונה של קובץ הוא האות המתאימה לו, ולאחר מכן מספר סידורי מ 0 עד 30520 (מספר האותיות).

חלק שלישי: בניית המודל בעזרת רשת קונבולוציה

לצורך יצירת הרשת, קראתי הרבה מאמרים באינטרנט, עשיתי הרבה ניסוי וטעיה ובסוף הגעתי למודל הבא:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 randomcolordistortion (Rand  (None, 64, 64, 3)        0
 omColorDistortion)

 conv2d (Conv2D)             (None, 64, 64, 64)        15616

 max_pooling2d (MaxPooling2D  (None, 32, 32, 64)       0
 )

 conv2d_1 (Conv2D)           (None, 27, 27, 32)        73760

 batch_normalization (BatchN  (None, 27, 27, 32)       128
 ormalization)

 max_pooling2d_1 (MaxPooling  (None, 13, 13, 32)       0
 2D)

 dropout (Dropout)           (None, 13, 13, 32)        0

 conv2d_2 (Conv2D)           (None, 11, 11, 64)        18496

 max_pooling2d_2 (MaxPooling  (None, 5, 5, 64)         0
 2D)

 dropout_1 (Dropout)         (None, 5, 5, 64)          0

 flatten (Flatten)           (None, 1600)              0

 dense (Dense)               (None, 128)               204928

 dropout_2 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 128)               16512

 dropout_3 (Dropout)         (None, 128)               0

 dense_2 (Dense)             (None, 5)                 645

=================================================================
Total params: 330,085
Trainable params: 330,021
Non-trainable params: 64
```

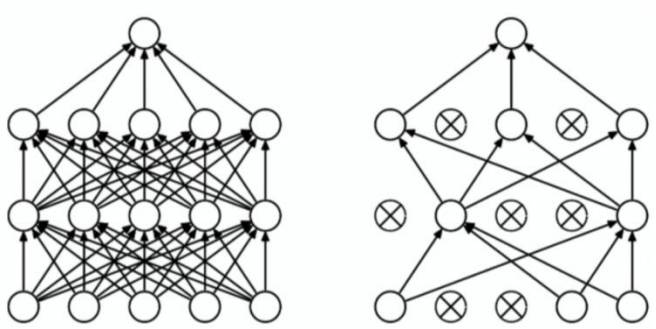
ברשת הקובולוצייה שיצרתי, השכבה הראשונה היא שכבת הData augmentation שתפורט בהמשך. הרשת מקבלת תמונות בגודל 64x64 פיקסלים ומעבירה אותם ברשת קובולוציה עם שכבות Maxpooling, Dropout,Batchnormalization, Flatten ולבסוף Dense.

קימפלתי את המודל עם אופטימייזר מסוג "Adam" הפופולרי, ההפסד מסוג "Cross Entropy" שנועד לבעיית הסיווג.

השתמשתי בטכניקת Learning rate decay שמקטינה את הלמידה כאשר המודל מגיע לשיא שלו בביצועיו ומאפשר צעדים קטנים יותר של למידה. בנוסף בכל epoch נשמר המודל הטוב ביותר.

כיצד מנעתי מ- overfitting:

- השתמשתי בטכניקת Data augmentation עלייה קראתי באינטרנט. שיטה זו מוסיפה לתמונה ניגודיות, בהירות, רוויה וגוון צבע כדי למנוע overfit לרשת. מימשתי טכניקה זו בכך שהוספתי שכבת נוירונים שיצרתי באמצעות מחלקה בשם "RandomColorDistortion".
- השתמשתי בטכניקת Dropout אשר "זורקת" באופן אקראי חלק מהנוירונים בעת אימון הרשת.



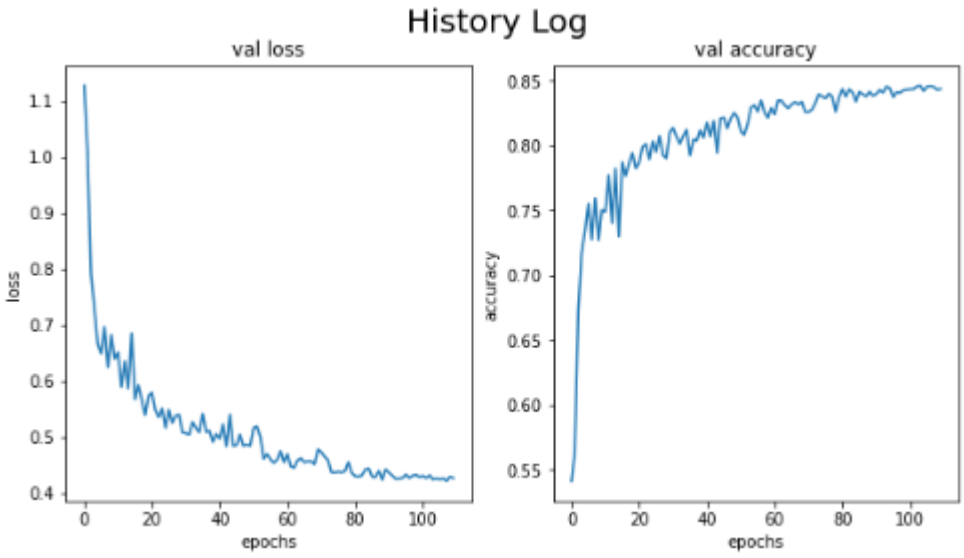
(a) Standard Neural Net (b) After applying dropout.

- עצירה מוקדמת של אימון הרשת, ברגע שהדיוק מפסיק להשתפר אין טעם להמשיך לאמן את הרשת.

הערה: ניסיתי להוסיף סיבוב, הזזה, זום אקראי, רעש גאוסני וטשטוש אך התוצאות היו פחות טובות מהמודל הנוכחי.

חלק רביעי: הערכת התוצאות

לאחר בערך 100 epochs הגעתי לדיוק של 0.843% על סט הולידציה.
להלן גרף ה- val loss ו val accuracy:



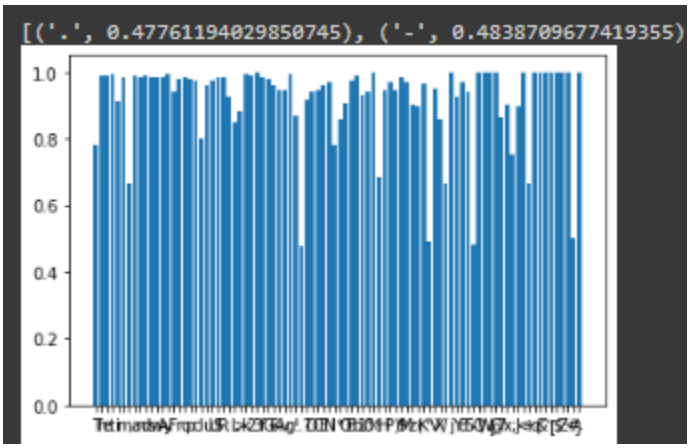
ניתן לראות שעצרתי את האימון בזמן לפני שהגעתי ל – overfit

חלק חמישי: אופטימיזציה

קיבלתי מודל טוב אשר יודע לסווג את הפונט של מעל 84% מהמילים וניסיתי לשפר אותו בהרבה דרכים.

השתמשתי בעובדה שכל מילה מורכבת מאותיות באותו פונט. כל תוצאה של המודל נותן וקטור הסתברויות לכל פונט, ולכן נוכל לסכום את כל הווקטורים בשביל תוצאה מקסימלית.

כמו כן, שמתי לב שלמודל יש תווים שהוא מסווג באופן מוצלח ויש תווים שהוא כמעט ולא מצליח לסווג. כדי לבדוק השערה זו חישבתי את אחוז ההצלחה של כל תו בנפרד ואכן ראיתי שאחוזי ההצלחה נעים בטווח הערכים 100% - 47% דיוק.



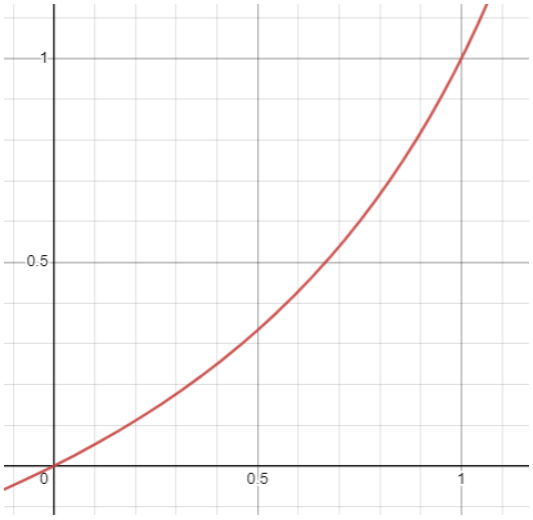
מהתבוננות, סיווג התו '.' עם הוא עם התוצאות הכי נמוכות(ככל הנראה אין הבדל משמעותי בין הפונטים) לעומת התו '3' אשר סווג תמיד כנכון.

לכן, בהינתן הפלט של המודל הנותן וקטור הסתברויות לכל פונט, החלטתי לכפול כל וקטור כזה בהסתברות ההצלחה של כל תו ורק אז לסכום את הווקטורים.

לאחר מכן שמתי לב שעבור תווים בעלי הסתברות גבוה המודל נותן יתרון גבוה מדי, ולכן השתמשתי בפונקציה הבאה בשביל לנרמל את וקטור ההסתברויות:

$f(x) = x / (2-x)$

(הגעתי לפונקציה זו באמצעות ניסוי וטעיה)



דוגמא להרצת האלגוריתם על מנת לסווג מילה:

- נגדיר מערך של אפסים באורך 5 (מספר הפונטים)
- נעבור בלולאה על אותיות המילה
- נעביר לרשת הקונבולוציה תמונה מעובדת של תו בגודל 64X64 פיקסלים
- נקבל וקטור באורך 5 כאשר כל תא המייצג את ההסתברות של כל פונט
- נכפול את הווקטור ב f(x) על מנת לנרמל את התוצאה
- נוסיף למערך שהגדרנו בהתחלה את הווקטור
- אחרי שעברנו על כל האותיות, התא בעל הערך הגדול ביותר הוא הפונט עם ההסתברות הגבוהה ביותר, נמיר את מספר התא לשם הפונט באמצעות הפונקציה "find_name" ובכך נקבל את הפונט המסווג למילה, וכמובן לכל אות בנפרד.

המשך חלק 5: הערכת תוצאות מחדש

לאחר שימוש באלגוריתם המתואר לעיל, קיבלתי את אחוזי הדיוק הבאים:

- 99.5% הצלחה על סט האימון
- 95.544% הצלחה על סט הולידציה

קיבלתי שיפור דרסטי באמצעות שימוש בשיטות המתוארות לעיל לעומת שימוש ברשת הקונבולוציה בלבד.

ניתן לראות שיש מעט overfit, אך עדיין התוצאות יחסית גבוהות בהינתן העובדה שיש הרבה תווים שחסר להם תמונות או שיש תווים שהפונט שלהם כמעט זהה.

אחוזי הצלחה לפי פונטים על סט הולידציה:

פונט	Alex Brush	Ubuntu Mono	Titillium web	Sansation	Open Sans
דיוק	0.99%	0.978%	0.957%	0.939%	0.873%

נשים לב שסיווג הפונט Open Sans הכי פחות מוצלח, אך זה לא מפתיע מכיוון שיש לפונט זה הכי פחות תמונות בסט האימון. לעומת זאת הפונט Alex Brush קיבל את התוצאה הטובה ביותר מכיוון שיש לו צורה ייחודית שקל להפריד משאר הפונטים.

סיכום:

הפרויקט היה מרתק ומלמד. למדתי המון בעיית סיווג התמונות, בפרט סיווג פונטים. הקדשתי הרבה זמן בעיקר על ניסוי וטעיה בבניית רשתות קונבולוציה עד שהגעתי לרשת הטובה ביותר, אך בסופו של דבר כל השלבים חשובים על מנת להגיע לתוצאה האופטימלית – מעיבוד מקדים על הנתונים ועד האופטימיזציה.
בסופו של דבר הצלחת המודל נמדדת לפי אחוז ההצלחה על סט הולידציה, והוא אכן היה יחסית גבוה - מעל 95%. לפי דעתי אם סט הנתונים היה גדול יותר ובנוסף הייתי מתמקד יותר בעיבוד מקדים יותר על סט האימון ואולי אפילו הוספת תמונות לסט, התוצאות היו יכולות להגיע למעל 98% ואף יותר.

תודה רבה על הקריאה :)