

פרויקט גמר

ב-deep learning computer vision

צביעת תמונות שחור-לבן

מגיש: יהונתן טל

בית ספר: מקיף י"א ראשונים

ת.ז: 324825470

מנחה: דינה קראוס

08.06.2020

תוכן עניינים

[מבוא 3](#_Toc40648430)

[פרטים כללים על התכנה 4](#_Toc40648431)

[מבנה הקוד 5](#_Toc40648432)

[מדריך למשתמש 6](#_Toc40648433)

[דרישות הפעלה 6](#_Toc40648434)

[מה נמצא בקובץ הפרויקט? 6](#_Toc40648435)

[הרצה תקינה של הפרויקט 6](#_Toc40648436)

[פלטים אפשריים 6](#_Toc40648437)

[מסלול הרצה 7](#_Toc40648438)

[מדריך למפתח 8](#_Toc40648439)

[קבצים 8](#_Toc40648440)

[תיאור פנימי של הקוד 9](#_Toc40648441)

[משתנים גלובלים 10](#_Toc40648442)

[מחלקות ופעולות עזר 11](#_Toc40648443)

[פעולות עזר לבדיקת הקבצים 14](#_Toc40648444)

[פעולות בדיקת המאגר וקביעתו 15](#_Toc40648445)

[פעולות שקשורות ליצירת המודל 17](#_Toc40648446)

[פעולות וגנרטורי יבוא מידע למודל 19](#_Toc40648447)

[פעולות אימון, הערכה, ושמירת המודל 22](#_Toc40648448)

[מסלול ההרצה 24](#_Toc40648449)

[רפלקציה 25](#_Toc40648450)

# מבוא

הסקריפט שפיתחתי הינו תכנה לצביעת תמונות בשחור לבן.

ספר זה הינו ספר הוראות למשתמש בתכנה. ישנו פירוט על המבנה של התכנה ומה שצריך כדי להפעיל אותה כמו שצריך. כמו גם, מה הן כל הפונקציות האפשריות בה ואיך אפשר להוציא ממנה את מלוא הפוטנציאל. בספר נמצא מדריך למשתמש וגם למפתח, לעיון של משתמשים המעוניינים להפעיל את התוכנית בהצלחה ולעיון מפתחים המעוניינים לשאוב השראה מהתוכנה. בספר נמצא גם פירוט על תהליך העבודה האישי שלי ומה הייתי צריך להשלים ולעבור כדי להגיע לתוצר הסופי, כל הקשיים וההצלחות הגדולות שלא צפיתי להן נכתבו בתמציתיות כדי לספק מענה לעוסקים בתחום המתמודדים עם אותן הבעיות.

תוכנית זו נכתבה כחלק מפרויקט הגמר במגמת מחשבים בבית הספר מקיף י"א ראשונים. נושא זה חדש בבית הספר שלנו ובכלל במערכת החינוך בארץ, לכן אני מעריך שהיה לנו קצת יותר קשה לעבוד על הפרויקט מאשר יורשינו בעתיד, שיוצעו להם מאגר עשיר יותר של חומרי לימוד נגישים ומנחים מומחים. התחלתי לכתוב את הסקריפט תחילה כתוכנה המסווגת בין תמונות לימונים על פי רמת הבשלות שלהן. בעת מחקר באינטרנט, נתקלתי בנושא צביעת התמונות והתעניינתי מאוד, החלטתי להחליף את הנושא שלי לזה ואינני מתחרט על כך כלל.

בעת המחקר שלי על הנושא, גיליתי שיש מספר תוכנות ושירותים אינטרנטיים חינמיים המציעים את השירות הזה ועושים עבודה טובה ביותר. מה גם שקיימת לאחרונה מגמת צביעת סרטים ישנים מתחילת המאה הקודמת באמצעות אותה הטכנולוגיה, וזאת בהצלחה רבה, ישנה אפילו פקודה בספרייה cv2 שעושה בדיוק את זה, היא מבוססת על אותם העקרונות של למידת מכונה.

הבעיה המרכזית שהייתה לי בעת כתיבת הפרויקט היא להתמודד עם העומס והכובד שיש להרצת התוכנית על המחשב. תהלך אימון מודל CNN הוא כבד ולוקח זמן רב, הוא דורש חישובים רבים מספור ומעמיס על המעבד (CPU) מאוד. פתרון אפשרי היה להשתמש במעבד גרפי (GPU), אך לא היה לי את הסוג המתאים לעבודה עם ספריית TensorFlow , עליה אפרט בהמשך, והדבר גלגל אותי לניסיונות שווא להתאים את המעבד הגרפי שברשותי לספרייה. לבסוף מצאתי פתרון באמצעות אתר Kaggle שמציע שירותי הרצה מרחוק לסקריפטים, בין היתר על מעבדים גרפים, בו כתבתי את רוב הסקריפט באמצעות עורך המבוסס על Jupiter Notebook, גם על כך אפרט בהמשך.

## פרטים כללים על התכנה

**מטרת התכנה**: ללמוד איך לצבוע תמונות שחור לבן (grayscale) באמצעות מאגר תמונות בצבע.

**בסיס תאורטי**:

* הלמידה נעשית באמצעות deep learning computer vision, שזהו סוג של Machine learning, למידת מכונה. זאת בעזרת מודלים המבוססים על רשתות עצביות מפותלות (CNN).
* נעשה שימוש רב בקוד בתבניות צבע (Color Formats/Spaces/Models) האלה RGB, RGBA, Grayscale, Lab:
  + RGB – red, green, blue תבנית צבעונית המורכבת משלושה צבעים המרכיבים את שאר הצבעים. הערכים הם מספרים שלמים הנעים בין 0 ל-255.
  + RGBA – red, green, blue, alpha תבנית צבעונית המורכבת משלושה צבעים ושקיפות, אלה מרכיבים את שאר הצבעים עם אפשרות לשקיפות ואטימות חלקית. הערכים הם מספרים שלמים הנעים בין 0 ל-255.
  + Grayscale – תבנית שחור לבן שבה הערכים מייצגים רק בהירות. הערכים הם מספרים לא שלמים הנעים בין 0 ל-1, עם 256 ערכים אפשריים.
  + LAB – lightness, green2red, blue2yellow תבנית צבעונית המורכבת משלושה טווחים המייצגים הבדלי צבעים. ערוץ L מייצג את הבהירות וערכיו מספרים שלמים הנעים מ-0 עד 100. ערוץ A מייצג את הטווח בין ירוק לאדום, וערוץ B מייצג את הטווח בין כחול לאדום, וערכיהם מספרים שלמים הנעים מ-(128-) עד 128.

**שפת התכנות והספריות:** שפת Python, שפה פופולרית להתעסקות עם DLCV. הספריות אשר התבצעו בעזרתן הלמידה הן TensorFlow ו-Keras, ספריות קוד פתוח ללמידת מכונה, המפותחת על ידי חברת גוגל לבנייה ואימון רשתות עצביות (NN).

**אופן ההפעלה:** המשתמש מריץ את התכנה והיא מדריכה אותו בין הפעולות השונות האפשריות באמצעות שאלות מרובות, עליהן המשתמש יכול לענות באמצעות הזנת תו או מילה.

# מבנה הקוד

הקוד מבצע את הלמידה על ידי קריאת המידע ממאגר המידע (Dataset) והעברתו לפעולת הלמידה של Keras. לפני כן מתבצעת בדיקה של מאגר המידע כדי למנוע תקלות בעת הרצת התוכנית. התוכנית מתבססת הרבה על תבניות צבע ולכן בדיקה זו היא קריטית למניעת טעויות המתמקדות בזה, שכן הדרך של התוכנית לפצל תמונות צבעוניות לשחור-לבן וצבע היא באמצעות פקודות המרה של תבניות צבע.

לאחר מכן מתבצע יבוא של מודל חיצוני בשם InceptionResnetV2 שהוא מודל שאומן מבעוד מועד על ידי אנשים אחרים בעולם למטרת סיווג תמונות. מודל זה הוא "חכם" ומורכב במיוחד ולכן השתמשתי בו באופן חלקי כדי לתת למודל שלי מושג יותר טוב לתוצאות שהוא אמור לייצר. בהמשך נוצר המודל שאני כתבתי, שמבוסס על CNN.

התוכנית ממשיכה לתהליך הלמידה באמצעות פקודת fit\_generator המשתמשת בפעולות גנרטורים (generators) שכתבתי כדי לייבא מידע לתהליך הלמידה בחתיכות קטנות, כדי לא להעמיס על זיכרון הגישה האקראית של המחשב (RAM).

לאחר מכן המשתמש יכול לבחור האם לשמור את המודל לאחר האימון והאם להציג גרף המציג את השתפרות המודל לאורך הזמן. בסוף, אם המשתמש מעוניין, ישנה אפשרות להציג את תוצאות האימון של הפרויקט באמצעות צביעת תמונות מהמאגר.

**הרחבה נמצאת במדריך למפתח.**

# מדריך למשתמש

## דרישות הפעלה

* על המשתמש להשתמש בסביבת עבודה המבוססת על Python 3.
* על סביבת העבודה של המשתמש להכיל את הספריות הבאות שלא מותקנות בברירת המחדל על Python 3:
  + tqdm
  + numpy
  + cv2
  + matplotlib
  + skimage
  + keras 2.2.4
  + tensorflow 1.11.0-rc1
* על המשתמש להיות בעל קובץ משקלים של InceptionResNetV2, שנמצא יחד עם מאגר התמונות ברירת המחדל.

## מה נמצא בפרויקט?

קובץ הפרויקט הוא קובץ zip שבו:

* הסקריפט של הפרויקט – קוד בשפת Python.
* מאגר התמונות עליו התנסיתי בעצמי (גרסת המאגר האינטרנטי flickr המכיל שלושים אלף תמונות).
* קובץ המשקלים של InceptionResNetV2 שפורסם רשמית על ידי Keras.

## הרצה תקינה של הפרויקט

הקוד שכתבתי עוצב במיוחד כדי להתמודד עם כל שגיאה שתקרה. שיניתי את התכנה במיוחד כדי להתמודד עם תבנית הצבעים RGBA בנוסף ל-RGB, ייתכן בהחלט שמישהו היה מייבא תמונות כאלה ואז הקוד שלי היה אובד עצות.

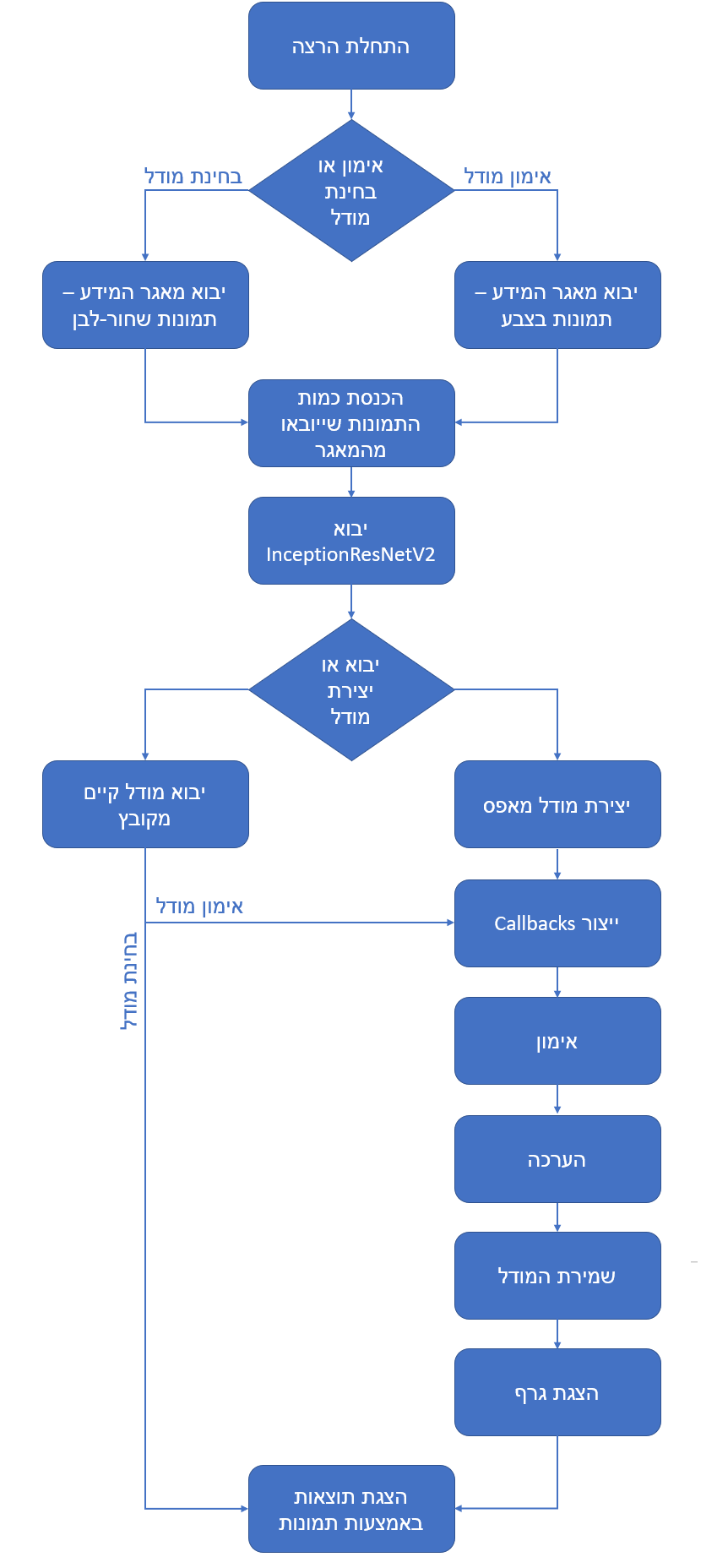
**השגיאה היחידה שלא נמצאה דרך לעקוף אותה היא חובת ההרצה של הפרויקט עם הגרסה הישנה יחסית של TensorFlow, 1.11.0-rc1 והגרסה הקודמת של Keras, 2.2.4.**

### פלטים אפשריים

התוכנית יוצרת מספר פלטים בהתאם לתפקידם, כל אחד בצבע אחר:

* סגול: בקשה מהמשתמש להכניס ערך מסוים.
* תכלת: מידע על תהליכי התוכנית. כלומר, יידוע המשתמש על תהליכים שהושלמו או דברים שהוא צריך לדעת לפני הפעלת פעולה כלשהי.
* אדום: אזהרה שנובעת מבעיה בקלט מהמשתמש.

### מסלול הרצה

זהו תרשים המתאר את מסלול ההרצה הכללי של התוכנית **מבחינת המשתמש**.

# מדריך למפתח

## קבצים

**Script.py** – הסקריפט של הפרויקט מכיל את כל מה שנמצא בארכיטקטורה וקצת יותר. הקובץ מכיל את כל המשתנים הגלובלים, המחלקות, והפעולות. זאת מבלי להשתמש בקבצים אחרים, רק בספריות החיצוניות שאפשר להתקין על סביבת העבודה. כתבתי את הסקריפט דרך העורך של אתר Kaggle שמבוסס על Jupiter Notebook, לכן הסקריפט מכיל תאי קוד (code cells), שחלק מהעורכים לא תומכים בהם (למשל PyCharm Community). תאי קוד אלה מאפשרים הרצה של חלקי קוד בנפרד ואני ממליץ מאוד על שימוש בהם.

פרטים נוספים: <https://jupyter.org/>

**flickr-image-dataset-compressed** – תיקייה שמכילה את מאגר התמונות לשימוש הסקריפט, 31,783 קבצי JPG. התמונות נלקחו מהאתר [flickr.com](https://www.flickr.com/) ומשמשות במקור למודלים שמתאימים משפטים באנגלית לתמונות כתיאור שלהן.

פרטים נוספים: <https://github.com/BryanPlummer/flickr30k_entities>

**keras\_\_inception\_resnet\_v2\_weights\_tf\_dim\_ordering\_tf\_kernels** – משקלי Inception שהורדתי מ-Kaggle והועלו רשמית על ידי Keras. InceptionResNetV2 הוא אחד מכמה מודלים שאומנו מראש שKeras- מציעים. כל אחד מהם נועד למטרה אחרת ונוצר ואומן על ידי אנשים אחרים מרחבי העולם.

פרטים נוספים: <https://keras.io/api/applications/>

## תיאור פנימי של הקוד

### משתנים גלובלים

אלו הם משתנים שהוגדרו כגלובלים בעת הגדרתם בתוך פעולות או מחוץ להן. הדבר אומר שהם משתנים שכל פעולה יכולה להשתמש בהם מבלי לייבא אותם. בהשראת סקריפטים רבים שראיתי באינטרנט החלטתי שאני מעדיף לעבוד עם משתנים מסוג זה. המשתנים הראשונים עד BATCH\_SIZE הוגדרו בגוף הראשי של הקוד, ומ- IMAGES\_PATHS המשתנים הוגדרו בתוך פעולות.

|  |  |  |
| --- | --- | --- |
| **תפקיד** | **סוג** | **שם** |
| זרע לאקראיות | Int | seed |
| מחליט על המסלול בו התכנה תתקדם: האם לייבא מאגר ולאמן מודל לפיו, או לייבא מודל קיים ולצבוע תמונות ממאגר | String | ACTION |
| רוחב התמונות איתן יעבוד המודל, לא צריך להיות קשור לרוחב האמיתי שלהן | Int | IMG\_WIDTH |
| אורך התמונות איתן יעבוד המודל, לא צריך להיות קשור לאורך האמיתי שלהן | Int | IMG\_HEIGHT |
| גודל החלקים הנפרדים (באצ'ים, batches) עליהן מתבצעת הלמידה | Int | BATCH\_SIZE |
| רשימה המכילה את הדרכים לכל הקבצים במאגר, לאחר מכן את כל הדרכים שמובילות לתמונות שאפשר לעבוד איתן | List | IMAGES\_PATHS |
| הדרכים לתמונות שנועדו לתהליך האימון | List | TRAIN\_PATHS |
| הדרכים לתמונות שנועדו לחלק הווידוא בתהליך האימון | List | VALIDATE\_PATHS |
| הדרכים לתמונות שנועדו לבדיקה הסופית של ביצועי המודל | List | TEST\_PATHS |
| כמות הערוצים של התמונות שאיתן התכנה תעבוד (0/1/3/4) בהתאם לתבנית הצבע המדוברת | Int | CHANNELS |
| כמות הפעמים שתהליך הלימוד יעבור על כל המאגר | Int | EPOCHS |
| המודל עצמו | Model | MODEL |
| מודל ה-InceptionResnetV2 | Model | INCEPTION |
| רשימה שבה מאוחסנים פעולות callbacks של Keras | List | CALLBACKS |
| תיעוד הלמידה לאורך הזמן, מתקבל מפעולת הלימוד | History | HIST |

### מחלקות ופעולות עזר

אלו הן פעולות שעוזרות לתפקוד התקין של הסקריפט, הן מטפלות בדברים קטנים שיהיה מיותר לכתוב כל פעם מחדש.

#### מחלקת Color

* מחלקה שמכילה מחרוזות שמייצגות הדפסה בצבעים בפייתון.

#### myprint

##### מקבלת:

* message – הודעה כמחרוזת.
* mode – מצב ההדפסה כתו.
* start\_newline – ערך בוליאני הקובע האם ליצור שורה חדשה בסוף ההדפסה.

##### מבצעת:

* מדפיסה את ההודעה בצבעים ש-Color מספקת, לפי mode.

##### מחזירה:

* כלום.

#### **path\_verify**

##### מקבלת:

* message – הודעה כמחרוזת.
* can\_be – רשימה המכילה מחרוזות של סוג הדרך (קובץ/תיקייה/זיפ).

##### מבצעת:

* מדפיסה את ההודעה באמצעות myprint.
* מקבלת מהמשתמש דרך כמחרוזת.
* מוודאת שהדרך תקינה ושהסוג שלה נמצא ב-can\_be.
* חוזרת על התהליך עד שיש דרך תקינה שאפשר להחזיר.

##### מחזירה:

* דרך כמחרוזת.

#### int\_verify

##### מקבלת:

* message – הודעה כמחרוזת.
* min\_ – מספר שלם מינימלי שאפשר להחזיר. ברירת מחדל: 1.
* max\_ – מספר שלם מקסימלי שאפשר להחזיר. ברירת מחדל: 1000000.

##### מבצעת:

* מדפיסה את ההודעה באמצעות myprint.
* מקבלת מהמשתמש קלט כמחרוזת.
* מוודאת שהקלט הוא מספר ושהוא בטווח בין min\_ ל-max\_.
* חוזרת על התהליך עד שיש מספר תקין שאפשר להחזיר.

##### מחזירה:

* מספר שלם.

#### option\_verify

##### מקבלת:

* message – הודעה כמחרוזת.
* options – רשימה של אפשרויות.

##### מבצעת:

* מדפיסה את ההודעה באמצעות myprint.
* מקבלת מהמשתמש קלט כמחרוזת.
* מוודאת שהקלט הוא ערך שקיים ב-options.
* חוזרת על התהליך עד שיש ערך תקין שאפשר להחזיר.

##### מחזירה:

* אפשרות כמחרוזת.

#### draw\_progress\_bar

##### מקבלת:

* n – מספר שלם, המיקום הנוכחי באיטרציה.
* total – מספר שלם, גודל האיטרציה.
* bar\_len – אורך ההדפסה.

##### מבצעת:

* מדפיסה באורך bar\_len סרגל התקדמות של איטרציה, כאשר n/total הוא האחוז הנוכחי.

##### מחזירה:

* כלום.

### פעולות עזר לבדיקת הקבצים

פעולות אלה הן פעולות שנעזרתי בהן כדי לבצע את בדיקת המאגר. הן מבצעות תהליכים טכניים שפישוטן סייע לי בכתיבת הקוד.

#### extraction

##### מקבלת:

* path – דרך, כמחרוזת.

##### מבצעת:

* אם הדרך מובילה לזיפ, מחלצת את הזיפ לדרך שתושג באמצעות path\_verify.

##### מחזירה:

* הדרך הנוכחית למאגר המידע, כמחרוזת.

#### scan\_dataset

##### מקבלת:

* path – דרך, כמחרוזת.

##### מבצעת:

* עוברת כל הקבצים בתוך התיקייה של הדרך ומצרפת את הדרכים שלהם לרשימה.

##### מחזירה:

* רשימה של כל הדרכים בתוך התיקייה של הדרך.
* אם הדרך היא קובץ, אז רשימה שמכילה את הדרך.

#### remove\_bads

##### מקבלת:

* unreadables – רשימה של דרכים.
* wrong\_colors – רשימה של דרכים.
* min\_ - מספר שלם.

##### מבצעת:

* עוברת על כל הדרכים בשתי הרשימות ומסירה אותן מ- IMAGES\_PATHS.
* מציעה אפשרות להדפיס את הדרכים האלה באמצעות myprint.

##### מחזירה:

* ערך בוליאני המעיד האם נשארו יותר מ-min\_ דרכים ב- IMAGES\_PATHS.

### פעולות בדיקת המאגר וקביעתו

פעולות אלה ממלאו את תפקיד יבוא המאגר וחילוקו לשם ביצוע תפקידים שונים בתכנה. הן מייבאות רק את החלקים הטובים של המאגר וכך מונעות בעיות עתידיות בהרצת הסקריפט.

#### normal\_check\_data

##### מקבלת:

* כלום.

##### מבצעת:

* מגדירה דרך למאגר באמצעות path\_verify. מחלצת עם extraction.
* מכניסה ל-IMAGES\_PATHS את כל הדרכים במאגר עם scan\_dataset.
* עוברת על הדרכים ב- IMAGES\_PATHSואם יש שם קובץ שהוא לא תמונה או תמונה שאי אפשר לקרוא אותה כמו שצריך, או תמונה עם מספר ערוצים (תבנית צבע) לא נכונה, **כולל תמונות grayscale**, מכניסה אותה לרשימות.
* CHANNELS נקבע לפי התמונה הראשונה שנמצאת.
* מסירה את הדרכים שברשימות מ- IMAGES\_PATHSבאמצעות remove\_bads.
* חוזרת על התהליך עד שישנו מאגר טוב שמכיל מספיק תמונות טובות.

##### מחזירה:

* כלום.

#### gray\_check\_data

##### מקבלת:

* כלום.

##### מבצעת:

* מגדירה דרך למאגר באמצעות path\_verify. מחלצת עם extraction.
* מכניסה ל-IMAGES\_PATHS את כל הדרכים במאגר עם scan\_dataset.
* עוברת על הדרכים ב- IMAGES\_PATHSואם יש שם קובץ שהוא לא תמונה או תמונה שאי אפשר לקרוא אותה כמו שצריך, או תמונה עם מספר ערוצים (תבנית צבע) לא נכונה, מכניסה אותה לרשימות.
* CHANNELS נקבע לפי התמונה הראשונה שנמצאת.
* מסירה את הדרכים הנ"ל מ- IMAGES\_PATHSבאמצעות remove\_bads.
* חוזרת על התהליך עד שישנו מאגר טוב שמכיל מספיק תמונות טובות.

##### מחזירה:

* כלום.

#### normal\_split\_data

##### מקבלת:

* כלום.

##### מבצעת:

* קולטת מהמשתמש את כמות התמונות מהמאגר איתו ירצה לאמן את המודל ולבחון אותו.
* חוזרת על התהליך עד שסכום הכמויות קטנן מסך הדרכים ב- IMAGES\_PATHSאו שווה לו.
* לפי כמויות אלה, מקצה דרכים אקראיות מ- IMAGES\_PATHSל-TRAIN\_PATHS, VALIDATE\_PATHS, TEST\_PATHS.
* קולטת מהמשתמש ערך ל- EPOCHS.

##### מחזירה:

* כלום.

##### **gray\_split\_data**

##### מקבלת:

* כלום.

##### מבצעת:

* אם IMAGES\_PATHS לא מכילה רק דרך אחת:
* קולטת מהמשתמש את כמות התמונות מהמאגר עליו ירצה לחזות את הצבע.
* חוזרת על התהליך עד שהכמות קטנה מסך הדרכים ב- IMAGES\_PATHSאו שווה לו.
* חותכת את IMAGES\_PATHS לגודל זה באקראיות.

##### מחזירה:

* כלום.

### פעולות שקשורות ליצירת המודל

פעולות אלה אחראיות לייצור/ייבוא המודל עצמו, איתו תתבצע הלמידה. המודל משלב בתוכו autoencoder ומסווג בשם InceptionResNetV2.

#### create\_inception

##### מקבלת:

* כלום.

##### מבצעת:

* אם INCEPTION לא מכילה קיים כבר:
* מקבלת דרך מהמשתמש עם path\_verify לקובץ של המשקלים של InceptionResNetV2, ימצא עם מאגר ברירת המחדל .
* יוצרת את InceptionResNetV2 וטוענת אליו את המשקלים.
* חוזרת על התהליך עד שהמשתמש הביא את הקובץ הנכון והמשקלים נטענו.
* משנה את תכונת ה-graph של INCEPTION לברירת המחדל של TensorFlow ב-session הנוכחי.

##### מחזירה:

* כלום.

#### colorize

##### מקבלת:

* כלום.

##### מבצעת:

* יוצרת מקודד אוטומטי (autoencoder) באמצעות CNN:
* יוצרת מקודד (encoder) באמצעות שכבות CNN שמקבל שכבת L של תמונת LAB.
* משלבת את המקודד עם קלט של הטבעה (embedding) של INCEPTION.
* יוצרת מפענח (decoder) על השילוב באמצעות שכבות CNN שמחזיר שכבות AB של תמונת LAB.

##### מחזירה:

* את המקודד האוטומטי כמודל.



#### model\_making

##### מקבלת:

* optimizer – המייעל (optimizer) של המודל שאיתו יתבצע העיבוד (compile).
* only\_load – ערך בוליאני הקובע האם צריך רק לטעון מודל קיים. ברירת מחדל: False.

##### מבצעת:

* אם only\_load או אם משתמש בוחר לטעון מודל קיים, טוען מודל ל-MODEL מדרך שמתקבלת עם path\_verify.
* חוזרת על התהליך עד שההעלאה מצליחה.
* אם משתמש בוחר ליצור מודל, MODEL נוצר בעזרת colorizeועובר עיבוד עם optimizer.
* אם המשתמש רוצה, מודפס תקציר המודל או נשמרת תמונה ויזואלית של המודל.
* בסוף הפעולה, MODEL הוא מודל שמקבל שכבת L של תמונת LAB והטבעה של INCEPTION ומחזיר שכבות AB של תמונת LAB.

##### מחזירה:

* כלום.

### פעולות וגנרטורי יבוא מידע למודל

אלה הן פעולות גנרטור שהשתמשתי בהם כדי לייבא מידע למודל בזמן האימון. פעולות גנרטור לא מחזירות (return), אלא מניבות (yield). כלומר, הן מחזירות ערכים בנפרד לאורך זמן. כל פעם שפונים אליהן, הן מניבות ערך אחד מתוך רבים באיטרציה מסוימת.

#### create\_inception\_embedding

##### מקבלת:

* grayscaled\_rgb – באץ' (מערך ארבע-ממדי) המכיל תמונות RGB שחור-לבן.

##### מבצעת:

* מתאימה את הבאץ' ל- INCEPTION.
* עם ערך ה-graph של INCEPTION כברירת המחדל של ה-session הנוכחי, מזינה את הבאץ' ל-INCEPTION ויוצרת תחזית (predict) שלו על בסיס הבאץ'.

##### מחזירה:

* תחזית של הבאץ' לפי INCEPTION.

#### train\_gen

##### מקבלת:

* כלום.

##### מבצעת:

* מתאימה את המשתנים ש- data\_transformer\_generatorמקבל לפעולת האימון של המודל.

##### מניבה:

* את הערכים מ-data\_transformer\_generator עם המשתנים המותאמים.

#### validate\_gen

##### מקבלת:

* כלום.

##### מבצעת:

* מתאימה את המשתנים ש- data\_transformer\_generatorמקבל לפעולת הווידוא בתוך האימון של המודל.

##### מניבה:

* את הערכים מ-data\_transformer\_generator עם המשתנים המותאמים.

#### test\_gen

##### מקבלת:

* is\_sample – האם הפעולה נקראה במיוחד כדי להניב ערכים לתהליך הצגת התוצאות. ברירת מחדל: False.

##### מבצעת:

* מתאימה את המשתנים ש- data\_transformer\_generatorמקבל לפעולות הבחינה של המודל.

##### מניבה:

* את הערכים מ-data\_transformer\_generator עם המשתנים המותאמים.

#### gray\_gen

##### מקבלת:

* כלום.

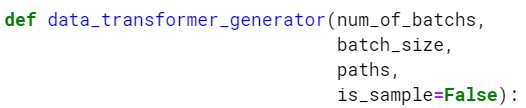
##### מבצעת:

* מתאימה את המשתנים ש- data\_transformer\_generatorמקבל לפעולת הבחינה של המודל עם תמונות שחור לבן.

##### מניבה:

* את הערכים מ-data\_transformer\_generator עם המשתנים המותאמים.

#### data\_transformer\_generator



##### מקבלת:

* num\_of\_batchs – מספר הבאצ'ים שהגנרטור יעבור עליהם.
* batch\_size – גודל הבאצ'ים הנ"ל.
* paths – הדרכים לתמונות.
* is\_sample – האם הפעולה נקראה במיוחד כדי להניב ערכים לתהליך הצגת התוצאות. ברירת מחדל: False.

##### מבצעת:

* מרכיבה כל באץ' מהתמונות שב- path, בהסתמך על תבנית הצבע של התמונות.
* מייצרת את ההטבעה בעזרת create\_inception\_embedding.
* הופכת את הבאץ' ל-RGB ומשם ל-LAB.

##### מניבה:

* אם is\_sample:
  + את הבאץ' כ-RGB
  + את הבאץ' כ-LAB
  + את התחזיות של המודל על שכבת ה-L וההטבעה.
* אם לא:
  + רשימה שמכילה את שכבת L של הבאץ' כ-LAB וההטבעה
  + שכבות AB של הבאץ' כ-LAB

### פעולות אימון, הערכה, ושמירת המודל

#### callbacks\_making

##### מקבלת:

* כלום.

##### מבצעת:

* מייצרת callback שמנמיך את שיעור הלמידה (learning rate/lr) פי 5 אם היא לא משתפרת לאחר 5 איפוצ'ים רצופים.
* אם המשתמש רוצה, מייצרת callback ששומר את המודלים הטובים ביותר (לפי קריטריון שהמשתמש מזין) בעת האימון.
* מכניסה את הערכים האלה ל-CALLBACKS

##### מחזירה:

* כלום.

#### training

##### מקבלת:

* כלום.

##### מבצעת:

* מאמנת את MODEL עם fit\_generator עם כל גנרטורים והערכים המתאימים.
* מדפיסה סרגלי התקדמות ונתוני דיוק (accuracy) והפסד (loss) בכל איפוץ'.
* מייצרת את HIST כמשתנה מסוג History שמכיל את המידע של האימון.

##### מחזירה:

* כלום.

#### evaluating

##### מקבלת:

* כלום.

##### מבצעת:

* מעריכה את MODEL עם evaluate\_generator עם כל גנרטורים והערכים המתאימים.
* מדפיסה סרגלי התקדמות ונתוני דיוק (accuracy) והפסד (loss).

##### מחזירה:

* כלום.

#### model\_saving

##### מקבלת:

* כלום.

##### מבצעת:

* אם המשתמש מעוניין, MODEL נשמר לכתובת מהמשתמש. מודל ומשקלים בנפרד.

##### מחזירה:

* כלום.

#### sampling\_graph

##### מקבלת:

* כלום.

##### מבצעת:

* אם המשתמש מעוניין, מציגה גרף של פרטי הלימוד (דיוקים, הפסדים ושיעור הלמידה) לאיפוץ', באמצעות HIST.

##### מחזירה:

* כלום.

#### sampling\_images

##### מקבלת:

* grays – האם הצגת התוצאות היא רק לתחזית המבוססת על תמונות שחור-לבן.

##### מבצעת:

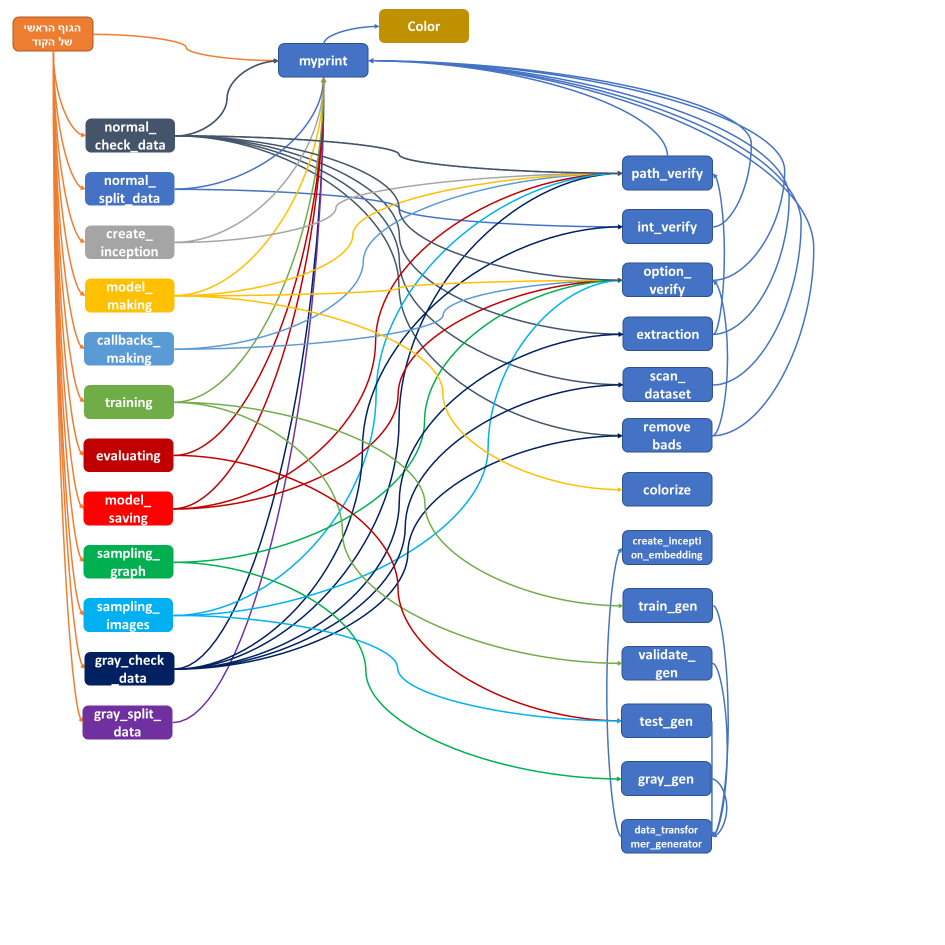
* אם המשתמש מעוניין, מקבלת ערכים מגנרטור מתאים (test\_gen / gray\_gen) ומשתמשת בהם כדי להציג תמונות שחור לבן, בצבע, ותחזיות צבע לפי המודל, זו לצד זו לשם השוואה.
* אם המשתמש מעוניין, התמונות עם הצבע שנחזה נשמרות.

##### מחזירה:

* כלום.

## מסלול ההרצה

זהו תרשים קריאות (call graph) המציג את הקריאות של כל פעולה לכל פעולה.



# רפלקציה

העבודה על הפרויקט עבורי לא הייתה קלה. כן האמנתי בהתחלה שהיא תהיה נורא קלה ושאני אפילו אסיים אותה עד ינואר, אך יש לציין שטעיתי טעות מרה. תחילה הייתי צריך לחקור הרבה דברים באינטרנט כדי להבין מושג קלוש על מה מדברים במקומות אחרים באינטרנט שהסבירו איך מבצעים את המשימה שבחרתי לעצמי – לסווג לימונים לפי רמת הבשלות שלהם. התנסיתי עם כל מיני קודים באינטרנט שעשו את אותו והתחוור לי יותר מרגע לרגע כמה חסר מושג אני.

אני לא גאה בכך אבל ניסיתי להתעלם כמה שיותר מחוסר הידיעות שלי וניסיתי כמה שיותר פקודות, פעולות, גנרטורים ומה לא שמצאתי באינטרנט בניסיונות עוורים אולי להצליח במשימה שלי. לאחר חודש או שניים בעת ניסויי היותר-מוצלחים עם קוד תכנת סיווג הלימונים שלי, הגעתי לשתי מסקנות קריטיות – שהפרויקט שלי זז לאט מאוד, ושהפרויקט שלי לא מעניין אותי.

בראשונה טיפלתי מיד באופן בלתי מודע, בעת המחקר שלי על אחד הנושאים, נתקלתי בסקריפט שצבע תמונות שחור לבן, בהצלחה מועטה יחסית. הדבר פתח אותי לנושא ועניין אותי במיוחד. שם הוסבר נושא ה-LAB וכל מיני פקודות שלא הכרתי. התחלתי להתמקד בנושא הזה, ולאחר שבועיים של מחקר וניסיונות, הבנתי שזה הכיוון שאני מעוניין להתמקד בו וידעתי את המורה (דינה קראוס) בכך שאני רוצה להחליף נושא.

עכשיו נותרה לי בעיה אחת, המחשב שלי לא עמד בציפיות. לא היה לי RAM מספיק גדול ו-CPU מספיק חזק כדי להריץ את הפרויקט ביעילות. תחילה ניסיתי לצמצם את העומס על ה- RAM באמצעות כיווץ (compression) התמונות ומחיקת משתנים תוך כדי ההרצה, אך הגעתי למסקנה שזה לא מספיק. בין היתר השתמשתי בספריות שונות כדי לעקוב אחרי ניצול הזיכרון, ויתכן שלמדתי מיומנויות שונות שיעזרו לי בעתיד.

מצד שני, הבנתי מהר מאוד מה יוריד את העומס מהמעבד שלי, מעבד גרפי. ואיזה יופי, יש לי שניים! רק חבל ששניהם לא מהחברה המתאימה, ולכן אי אפשר לעבוד איתם... לאחר ניסיונות שווא רבים מספור לגרום לזה לעבוד איכשהו, הייתי נורא קרוב ללהתייאש לגמרי ואף הפסקתי לעבוד על הפרויקט לחודשיים. המזל שלי הוא שאני מכיר סטודנט אחד שבדיוק מסיים לימודי הנדסת מערכות מידע שהכיר לי את פונקציית העריכה וההרצה ב-Kaggle. שם יכולתי להריץ את הפרויקט אין-ספור פעמים לאורך זמן רב על מעבד גרפי, ביעילות ובמהירות.

התמודדתי עם בעיית הזיכרון בסופו של דבר לאחר שפתרתי את בעיית המעבד, עשיתי זאת עם שימוש בפעולות גנרטורים. אני חייב להביע צער על כך שלא מלמדים עליהן בבית הספר משום שכעת הן נראות לי חלק אינטגרלי מכתיבת סקריפט בפייתון, ואני בטוח שהיכולת שלי להשתמש בהן תעזור לי בעתיד. היכולת של גנרטורים להניב ערכים לאורך זמן בלי להעמיס על הזיכרון הוציאה אותי מהבעיה הזו והפכה את הפרויקט שלי למסוגל להתמודד עם כל כמות של תמונות, לא משנה כמה גדולה.

Kaggle מגדירים את עצמם כ-"בית שלך למדע נתונים". ואני לא יכול לחלוק עליהם כלל, רק בזכותם הצלחתי להתקדם עם הקוד ולכתוב הרבה דברים אחרים ששיפרו אותו בלי להתעסק במגבלות הטכניות של המחשב שלי. מה גם שהאתר מציע מאגרי מידע ותמונות רבים מספור שאפשרו לי להתנסות בהמון מצבים שונים ולעצב תוכנית יעילה במיוחד.

אם הייתי מתחיל לעבוד על הפרויקט כיום, אני יודע שהייתי עושה עבודה הרבה יותר טובה. ראשית, אני מכיר את הנושא הרבה יותר טוב ואני מיודע ברוב הנושאים של כתיבת תוכנית ללימוד רשת עצבית. הנושאים האלה הם נורא קשים ללמידה ללא מדריך ובשפה שהיא לא שפת אם. אני כן מאמין שהתמודדתי עם האתגר בהצלחה, אבל אין לדעת לאילו אופקים הייתי יכול להגיע אם הייתי מתחיל את העבודה עם הידע שיש לי כבר בנושא והפנאי ללמוד עוד.

שנית, עכשיו אני מכיר את Kaggle שהייתה מקלה עליי מאוד בתחילת הדרך, משום שהיא גם מכילה מדריכים והמון המון דוגמות בנושא הפרויקט. ישנן רבבות של משתמשים באתר שיכולים לענות על שאלות ולהציע פתרונות ובכך להקל על תהליך העבודה של יחיד בהרבה. מיותר לציין שעם העורך המתקדם שלהם והיכולת להריץ תוכניות על חומרה חיצונית בחינם, הדבר מזרז בהרבה את הכתיבה עצמה ומקל על חוסר סבלנות ומשברים בפן הרגשי שיכולים להגרם בעת העבודה על נושא מורכב כמו זה.