פרויקט גמר

ב- deep learning
למידת צביעת תמונות
שחור-לבן

מגיש: יהונתן טל

בית ספר: מקיף י"א ראשונים

ת.ז: 324825470

מנחה: דינה קראוס

06.2020

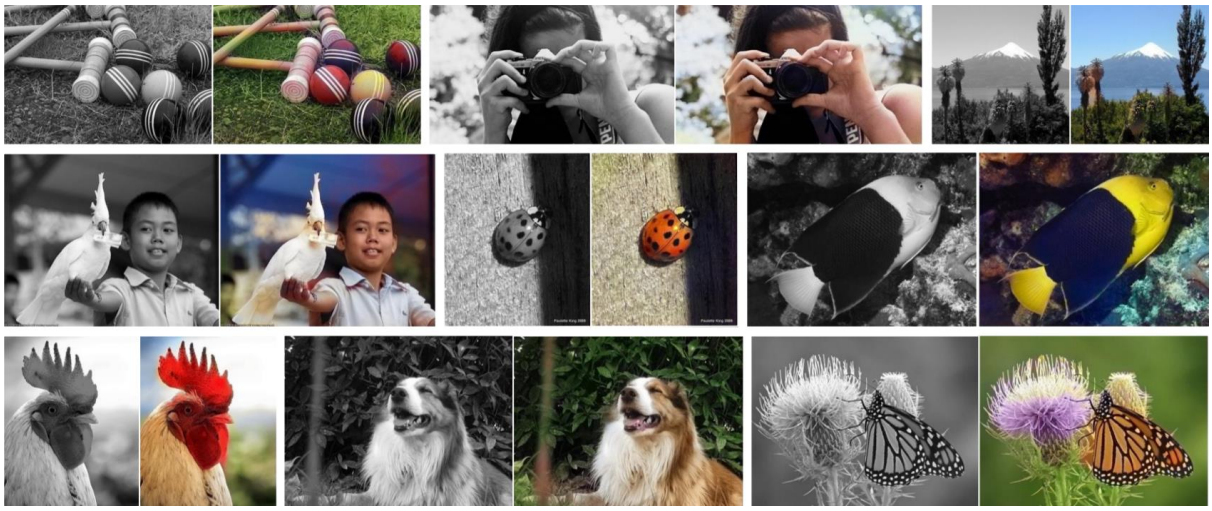
תוכן עניינים

מבוא

הסקריפט שפיתחתי הינו תכנה לצביעת תמונות בשחור לבן. המודל שמבצע את הלמידה מקבל תמונה מכל סוג שתבנית הצבע שלה היא grayscale (שחור-לבן), ומחזיר תמונה עם תבנית צבע RGB שאמורה להיות צבועה בצבע אמין. על תבניות הצבע (RGB, Grayscale) יפורט בהמשך.

ברור כי הקשר בין הגרסה השחורה-לבנה לבין הגרסה הצבועה של תמונה היא כללית למדי, כפי שאפשר לראות בתמונות מטה. זו הסיבה המרכזית לכך שהגעה לתוצאות טובות עם הבעיה הזו זה מאתגר במיוחד. בתמונות הבאות לדוגמה, אם מסתכלים על התמונות השחורות-לבנות בלבד, ניחוש הצבעים המדויקים של התמונה המקורית נראה בלתי-אפשרי. יתרה מכך, מביני עניין יודעים ששליש מהמידע בתמונה נאבד כשהיא מצולמת בשחור-לבן (מימד אחד במקום שלושה).

עם זאת, במבט מקרוב ניתן להבחין כי במקרים רבים הסמנטיקה של הסצנה בתמונה מספקת רמזים בשפע לצביעת חלקים רבים בכל תמונה: הדשא הוא ירוק, השמיים כמעט תמיד כחולים, והפרת-משה-רבנו בהחלט אדומה. כמובן, תחזיות סמנטיות מסוג זה אינן תקפות לכל דבר. למשל, כדורי הקריקט שעל הדשא עשויים להיות בצבעים אחרים משניתנו להם. עם זאת, מטרתי אינה בהכרח לשחזר את הצבע המקורי בפועל, אלא לייצר צבע מתקבל על הדעת שיכול להטעות את הצופה האנושי. לפיכך, המשימה שלי הופכת להיות הרבה יותר ברת-השגה: לייצר תלות סטטיסטית בין סמנטיקה של תמונות שחור-לבן לבין הגרסות הצבועות שלהן, על מנת להביא לתוצאות חזותיות משכנעות.



ספר זה הינו ספר הוראות למשתמש בתכנה או למפתח הבא להבינה לעומק ו/או לשאוב השראה ממנה. 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 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.

פרטים כללים על התכנה

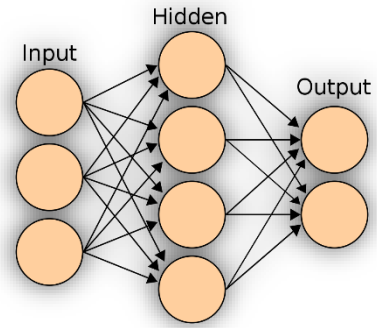
מטרת התכנה: ללמוד איך לצבוע תמונות שחור לבן (grayscale) באמצעות מאגר תמונות בצבע. פירוט בבסיס התיאורטי.

שפת התכנות והספריות: שפת Python 3. הספריות אשר התבצעו בעזרתן הלמידה הן TensorFlow ו-Keras, ספריות קוד פתוח ללמידת מכונה, המפותחת על ידי חברת גוגל לבנייה ואימון רשתות עצביות. נעשה שימוש במספר ספריות נוספות לעיבוד המידע והתמונות ולהצגתם, יפורט עליהן בהמשך.

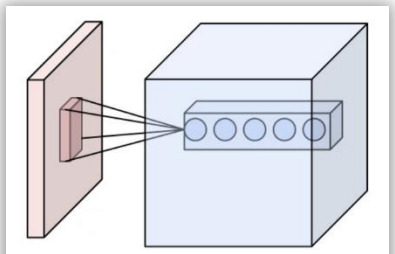
אופן ההפעלה: המשתמש מריץ את התכנה והיא מדריכה אותו בין הפעולות השונות האפשריות באמצעות שאלות מרובות, עליהן המשתמש יכול לענות באמצעות הזנת תו או מילה.

בסיס תאורטי

הלמידה נעשית באמצעות deep learning, שזהו סוג של Machine learning, למידת מכונה. זאת בעזרת מודלים המבוססים על רשתות עצביות מפותלות (CNN).



רשת עצבית (NN) היא מודל מתמטי חישובי שפותח בהשראת תהליכים מוחיים או קוגניטיביים המתרחשים ברשת עצבית טבעית ומשמש במסגרת למידת מכונה. רשת מסוג זה מכילה בדרך כלל מספר רב של יחידות מידע (קלט ופלט) המקושרות זו לזו, קשרים שלעיתים קרובות עוברים דרך יחידות מידע "חבויות" (Hidden Layer). צורת הקישור בין היחידות, המכילה מידע על חוזק הקשר, מדמה את אופן חיבור הנוירונים במוח.

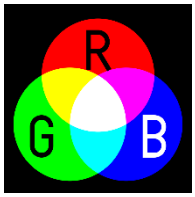


נוירונים של שכבה מפותלת (בכחול) מחוברת לשדה הקבלה של השכבה הקודמת (באדום)

רשת עצבית מפותלת (CNN) היא סוג של רשת עצבית, המיושמת לרוב לניתוח דימויים חזותיים, כמו תמונות וסרטונים. אלה מקבלות כרכים תלת ממדיים (רוחב, גובה ועומק) של נוירונים. כאשר כל נוירון בתוך שכבה מפותלת מחובר רק לאזור קטן בשכבה שלפניו, שנקרא שדה קבלה.

עם שליחת מספיק תמונות לרשת עצבית מפותלת, היא אמורה לבצע חיזוי מתמטי של הערכים שאמורים לייצג תמונה (במקרה שלי) לפי תמונה אחרת.

נעשה שימוש בקוד **בתבניות צבע** (Color Formats/Spaces/Models) שהן בעצם ייצוג מספרי לדרך בה מוגדר צבעו של כל פיקסל במסך בתצוגת המחשב.



- **RGB** – red, green, blue – תבנית צבעונית המורכבת משלושה צבעים המרכיבים את שאר הצבעים. הערכים הם מספרים שלמים הנעים בין 0 ל-255. משומשת בעיקר על ידי תמונות מסוג JPEG/JFIF/JPG.

- **RGBA** – red, green, blue, alpha – תבנית צבעונית המורכבת משלושה צבעים ושקיפות, אלה מרכיבים את שאר הצבעים עם אפשרות לשקיפות חלקית. הערכים הם מספרים שלמים הנעים בין 0 ל-255. משומשת בעיקר על ידי תמונות מסוג GIF/PNG.
- **Grayscale** – תבנית שחור-לבן שבה הערכים מייצגים רק בהירות. הערכים הם מספרים לא שלמים הנעים בין 0 ל-1 עם 256 ערכים אפשריים, או מספרים שלמים שנעים בין 0 ל-255. לא משומש בדרך כלל משום שתמונות נשמרות כמעט תמיד כ-RGB או כ-RGBA.

יהונתן טל

מדריך למשתמש

פה נמצא המדריך המעשי למשתמש בתכנה, כל מה שהוא צריך לדעת כדי להפעיל אותה בצורה הטובה והיעילה ביותר.

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:
  - NumPy
  - Matplotlib
  - Skimage (scikit-image)
  - Keras
  - TensorFlow

התקנת הספריות יכולה להתבצע באמצעות pip (מתקין הספריות של Python) באופן הבא (עם אותיות קטנות בלבד):

```
pip install library
```

אם רוצים להוריד גרסה ספציפית (יתכן עם TensorFlow ו-Keras) מוסיפים == ואת מספר הגרסה (גם כאן עם אותיות קטנות בלבד):

```
pip install library==version
```

אם מבצעים את ההתקנה דרך פלטפורמות המבוססות על Jupiter Notebook, מוסיפים סימן קריאה לפני (גם כאן עם אותיות קטנות בלבד):

```
!pip install library
```

הרצה תקינה של הפרויקט

התחלת ההרצה

- הרצה דרך פלטפורמות מבוססות על command line כמו command prompt תעשה בצורה הבאה:

```
(base) C:\WINDOWS\system32>python \path...\Project\Script.py
```

- דרך פלטפורמות המבוססות על Jupiter Notebook אפשר פשוט להריץ על ידי לחיצה על הכפתור הרלוונטי לאחר טעינת הקוד, כמחברת או כסקריפט.

תיקיות פלט

התוכנית שומרת את כל הקבצים שהמשתמש מעוניין בהם בתיקייה בשם OUTPUT בתוך התיקייה שבה נמצא הסקריפט.

- מודלים והצגה כתמונה של המודל, נשמרים בתת-תיקייה models
- תוצאות כתמונות, נשמרות בתת-תיקייה results

מקרי קיצון

- אם המאגר ריק או לא נמצא, התוכנית תבקש מהמשתמש לבחור אחד חדש.
- אם כל סוג של קלט אינו תקין, כמו דרך (path) למאגר, כמות תמונות, דרך למודל, וכד' – מודפסת אזהרה, כמו שמפורט באזהרות אפשריות.

פלטים אפשריים

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

- סגול: בקשה מהמשתמש להכניס ערך מסוים.
- ציאן: 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.

הרצה מבחינת המשתמש

ראשית, התוכנית מבקשת לבחור מסלול הרצה:

- **1 – לאימון מודל על מאגר תמונות.**
- **2 – לחזיית תמונות ממאגר לפי מודל קיים.**

```
Hello stranger, this is my python based deep learning project designed to learn how to color black and white images.
What would you like to do?
1 - model training
2 - predicting color of BW images with existing model
```

יהונתן טל

התוכנית שואלת האם להשתמש במאגר ברירת המחדל. אם כן, מציגה פרטים על דרישות התוכנית ומתחילה את הבדיקה.

```
Use default dataset? [y/n]
y

Due to this deep learning project's purpose, any grayscale image will be excluded.
Any non-image format or image with broken data will also be excluded.
The program works with ONE color format (RGB/RGBA) per run for all the images you choose, based of your first image. Any other
color format will not be accepted.
...
Loaded 32657 files from dataset.
Checking dataset for incorrect files or images...
[                                                  ] 0.14%
```

אם לא, התוכנית מבקשת דרך (path) חלופית למאגר, ומשם הכל זהה. **אם בחרנו במסלול 2:** חשוב שנכניס מאגר עם תמונות שחורות-לבנות.

```
Use default dataset? [y/n]
n

Due to this deep learning project's purpose, any grayscale image will be excluded.
Any non-image format or image with broken data will also be excluded.
The program works with ONE color format (RGB/RGBA) per run for all the images you choose, based of your first image. Any other
color format will not be accepted.
Enter path to dataset. Can be zip or dir:
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/
...
Loaded 2128 files from dataset.
Checking dataset for incorrect files or images...
[=====                                             ] 10.57%
```

אם יש קבצים עם בעיות, לפי ההסבר בדרישות התוכנית, שואל אם להדפיס את הדרכים (paths) שלהם. אם כן, מדפיס. (תמונה חלקית)

```
86 wrong format files or unreadable images.
Print paths? [y/n]
y
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/1000.jpg
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/1475.jpg
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/1375.jpg
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/0225.jpg
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/0825.jpg
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/1400.jpg
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/0650.jpg
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/1900.jpg
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/0900.jpg
../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/0400.jpg
```

אם אין, לא מבקש.

```
0 wrong color format excluded images.
```

בסוף מדפיס כמה תמונות נותרו.

```
2042 images are fine.
```

לאחר מכן, התוכנית מבקשת כמויות לתהליכי האימון והבחינה, ומספר החזרות על המאגר (epochs) שיתרחשו.

```
Enter how many images to take for training proccess, including validation in 2:8 ratio (hundreds~thousands):
2000
Enter how many images to take for testing proccess (singles):
20
Paths randomly assigned for training, validating and testing.

Enter how many epochs the training process will take (aprox. 1k images per epoch: GPU-1.5m | CPU-0.5h):
5
```

אם בחרנו במסלול 1: שואל האם ליצור או לייבא מודל קיים.

אם בחרנו ליצור, נוצר מודל ויש לנו אפשרות לשמור תצוגה ויזואלית שלו כתמונה (ישנו עותק בנספחים) /להדפיס את הפרטים של כל השכבות שלו.

```
Would you like to load an existing model or create a new one? [load/create]
create
...
Model created and compiled.
Would you like to:
1. save visualiztion of model as an image.
2. print summary of model.
3. do nothing.
```

אם בחרנו לייבא, מבקש דרך (path) למודל ומייבא אותו.

```
Would you like to load an existing model or create a new one? [load/create]
load
Enter path for model. Can be file:
../input/flickr-image-dataset-30k/Colorization_Model.h5
Would you like to:
1. save visualiztion of model as an image.
2. print summary of model.
3. do nothing.
```

אם בחרנו במסלול 2: מבקש דרך (path) למודל ומייבא אותו.

```
Enter path for model. Can be file:
../input/flickr-image-dataset-30k/Colorization_Model.h5
Would you like to:
1. save visualiztion of model as an image.
2. print summary of model.
3. do nothing.
```

יהונתן טל

אם לוחצים 1, התמונה נשמרת ומודפס המקום בו נשמרה.

```
1
Saved visualiztion at:  /kaggle/working/models/Colorization_Model-visualization.png
```

אם לוחצים 2, מודפס תקציר של המודל. (תמונה חלקית)

```
2
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_9 (InputLayer)            (None, 256, 256, 1)  0
__________________________________________________________________________________________________
conv2d_252 (Conv2D)             (None, 256, 256, 128 1280        input_9[0][0]
__________________________________________________________________________________________________
max_pooling2d_14 (MaxPooling2D) (None, 128, 128, 128 0           conv2d_252[0][0]
__________________________________________________________________________________________________
conv2d_253 (Conv2D)             (None, 128, 128, 128 262272      max_pooling2d_14[0][0]
__________________________________________________________________________________________________
conv2d_254 (Conv2D)             (None, 128, 128, 128 147584      conv2d_253[0][0]
__________________________________________________________________________________________________
max_pooling2d_15 (MaxPooling2D) (None, 64, 64, 128)  0           conv2d_254[0][0]
__________________________________________________________________________________________________
conv2d_255 (Conv2D)             (None, 64, 64, 256)  524544      max_pooling2d_15[0][0]
__________________________________________________________________________________________________
```

אם בחרנו במסלול 1: התוכנית שואלת האם לשמור את המודל בעת הריצה של האימון. אם כן, בוחרים לפי איזה קריטריון.

```
Save best model while runnig? [y/n]
y
By which monitor? [loss / acc / val_loss / val_acc]
val_loss
```

אם בחרנו במסלול 1: הלימוד מתחיל. (תמונה חלקית)

```
Beginning training...
Epoch 1/25
62/62 [==============================] - 497s 8s/step - loss: 0.0138 - acc: 0.6018 - val_loss: 0.0127 - val_acc: 0.6245
Epoch 2/25
62/62 [==============================] - 447s 7s/step - loss: 0.0137 - acc: 0.6022 - val_loss: 0.0127 - val_acc: 0.6246
Epoch 3/25
62/62 [==============================] - 455s 7s/step - loss: 0.0136 - acc: 0.6022 - val_loss: 0.0127 - val_acc: 0.6245
Epoch 4/25
62/62 [==============================] - 464s 7s/step - loss: 0.0136 - acc: 0.6022 - val_loss: 0.0126 - val_acc: 0.6244
```

אם בחרנו במסלול 1: הערכה מתבצעת, מודפסים הערכים שיצאו בסוף.

```
5/5 [==================================] - 5s 971ms/step
loss: 0.0036, accuracy: 0.8850
```

אם בחרנו במסלול 1: המשתמש נשאל אם לשמור את המודל לאחר הלימוד. אם כן, שומרת.

```
Save model after training? [y/n]
y
```

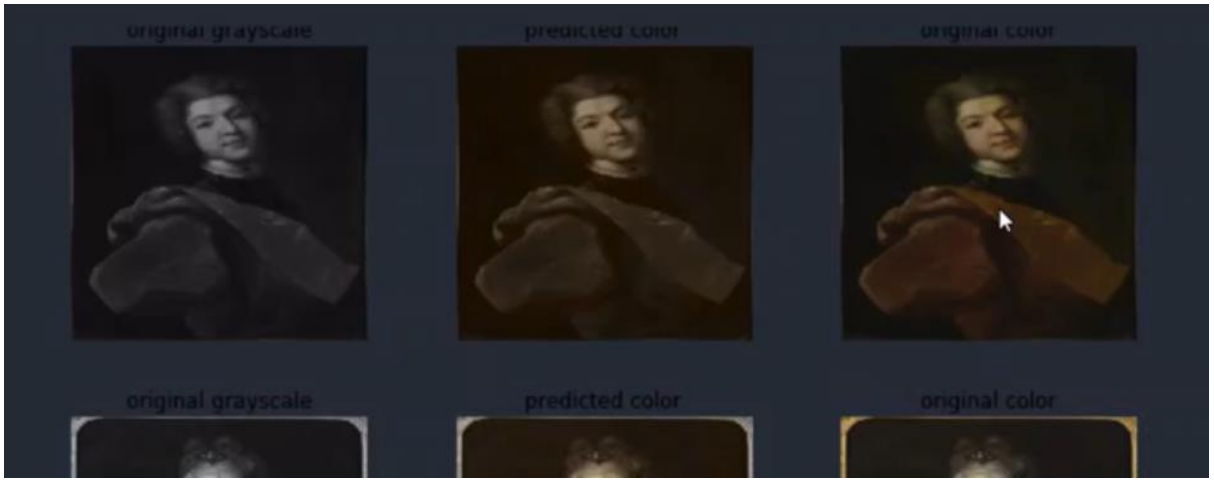
אם בחרנו במסלול 1: מציגה את הגרפים. (תמונה חלקית – תמונות מלאות במסקנות הפרויקט)

```
Show metrics graph? [y/n]
y
Save metrics graph? [y/n]
y
```

התוכנית שואלת את המשתמש אם הוא רוצה לראות את התוצאות ואם כן, האם הוא רוצה לשמור אותן. אם כן, ישמרו התמונות עם הצבע החזוי.

```
Show results of model colorization? [y/n]
y
Save results of model colorization? [y/n]
y
```

אם בחרנו במסלול 1: יודפסו 3 טורים של תמונות: צבע מקור, צבע חזוי, ושחור-לבן. (תמונה חלקית)



אם בחרנו במסלול 2: יודפסו 2 טורים של תמונות: צבע חזוי ושחור-לבן. (תמונה חלקית)

אזהרות אפשריות

לאחר כל אזהרה, המשתמש מתבקש לנסות שוב, עד שהוא מצליח.

במקרה של קלט דרך (path) לא נכונה או מסוג לא נכון:

```
Enter some path: Can be zip or file or dir:
acsjnlasc
Path is not zip/file/dir
```

במקרה של קלט תו שהוא לא מספר אם בוקש מספר:

```
Enter integer:
p
Input is not an integer.
```

במקרה של קלט מספר גדול מדי:

```
Enter integer:
200
Enter a smaller number.
```

במקרה של קלט מספר קטן מדי:

```
Enter integer:
2
Enter a larger number.
```

במקרה של קלט שלא מתאר אפשרות בשאלה מרובת אפשרויות:

```
yes or no? [y/n]
j
Not an option.
```

במקרה שהתוכנית לא מוצאת את המאגר ברירת המחדל, למרות שבוקשה לייבא משם קבצים:

```
Default dataset wan't found.
```

במקרה של מאגר ריק:

```
0 images are fine.
You have too little valid images in here, select another dataset.
```

במקרה של קלט כמויות גדולות מדי למאגר:

```
Enter how many images to take for training proccess, including validation in 2:8 ratio (hundreds~thousands):
2500
Enter how many images to take for testing proccess (singles):
50
You entered higher values than dataset provides (2042), try again.
```

במקרה של קלט דרך למודל שלא מובילה למודל שלם:

```
This model is weights only or incompatible.
```

יהונתן טל

מדריך למפתח

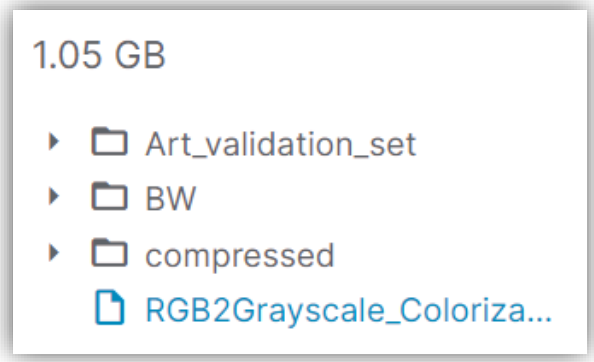
קבצים

הפרויקט נמצא באתר GitHub ומכיל 4 קבצים כמו בתמונה:

NewerScript.py	added cross-validation	2 days ago
README.md	Update README.md	43 minutes ago
Videorout1+2v3-comp.m4v	Added book and videoV3	7 days ago
ספר.pdf	Create ספר.pdf	41 minutes ago

NewerScript.py – הסקריפט של הפרויקט. מכיל את כל מה שנמצא בתיאור הפנימי. הסקריפט מכיל את כל המשתנים הגלובליים, המחלקות, והפעולות. כלומר, התוכנה פועלת ללא שימוש בקבצים אחרים. זאת מלבד הקבצים של הספריות 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). אלה הם שברי קוד שאפשר להריץ באופן עצמאי ללא תלות כמעט בשאר הקוד, תכונה שמעולה לניפוי באגים.

README.md – קובץ README שכתובים בו פרטים כלליים מאוד על הפרויקט וקישור למאגר המידע באתר Kaggle בשם my-colorization-dataset:



my-colorization-dataset – תיקייה שמכילה את מאגר התמונות ברירת המחדל לשימוש הסקריפט, בה 3 תיקיות:

- **compressed** – 31,783 קבצי JPG. התמונות נלקחו מהאתר flickr.com ומשמשות במקור למודלים שמתאימים משפטים באנגלית לתמונות כתיאור שלהן.
- **Art_validation_set** – 866 קבצים בתוך מספר תת-תיקיות ובסוגים מרובים. נלקח מהמאגר Art Images ב-Kaggle כדי לגוון את המאגר (להוסיף לו קבצים פגומים וסוגי תבניות צבע אחרים).
- **BW** – 7 תמונות שחור-לבן בתבניות Grayscale, RGB למטרת החיזוי. נלקחו מ-Google Images באקראיות.

Videorout1+2v3-comp.m4v – סרטון המכיל צילום מסך מתמשך של הרצת התוכנית דרך Kaggle.

ספר.pdf – ספר זה.

תיאור פנימי של הקוד

פרק זה מכיל את כל תוכן הקוד עם הסברים על כל משתנה ועל כל פעולה עם חשיבות לכלל התוכנית.

משתנים גלובליים

אלו הם משתנים שהוגדרו בתוך פעולות או מחוץ להן כגלובליים. הדבר אומר 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 עד סוף הטבלה המשתנים הוגדרו בתוך פעולות.

תפקיד	סוג	שם
מספר המשמש את הפעולות שמייצרות רצפים אקראיים למטרות התוכנית.	Int	seed
דרך (path) למיקום במחשב/בפלטפורמה בו נשמרים הקבצים. לדוגמה, אם הערך שווה ל- "D:\OneDrive" אז כל הקבצים ישמרו בתיקייה זו.	String	OUTPUT
הערך שלו מתקבל מהמשתמש. • אם הוא '1': התוכנית תייבא מאגר תמונות ותאמן מודל לפיו. • אם הוא '2': התוכנית תייבא מודל קיים ותצבע תמונות ממאגר תמונות שהיא תייבא.	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 (paths) לכל הקבצים במאגר. לאחר מכן מכילה את כל הדרכים מתוך אלה, שמובילות לתמונות שאפשר לשלוח למודל (מסלול 1) או לחזות צבע מהן (מסלול 2). ישנו צורך בסינון שכזה משום שישנן תמונות עם נתונים פגומים שאי אפשר להשתמש בהן, ובמסלול 1 אי אפשר להשתמש בתמונות Grayscale מן הסתם.	List	IMAGES_PATHS

תפקיד	סוג	שם
הדרכים לתמונות שנועדו לבדיקה הסופית של ביצועי המודל (מסלול 1) או לתמונות שנועדו לחיזוי צבע (מסלול 2).	List	TEST_PATHS
כמות הערוצים (channels) של התמונות שאיתן התכנה תעבוד (0/1/3/4) בהתאם לתבנית הצבע המדוברת (RGB/RGBA/Grayscale). לפי מספר זה נעשים סינון התמונות לתמונות שאפשר להשתמש בהן, ושליחת התמונות למודל בצורה הנכונה (משום שהוא מקבל רק תמונות עם 3 ערוצים).	Int	CHANNELS
כמות הפעמים (epochs איפוצ'ים) שתהליך הלימוד יעבור על כל המאגר.	Int	EPOCHS
המודל עצמו.	Keras.Model	MODEL
תיעוד הלמידה לאורך הזמן, מתקבל מפעולת הלימוד.	Dictionary	HIST
זמן ההתחלה בזימון פעולת draw_progress_bar.	Time.time	start

גוף הקוד הראשי

מבצע:

- מגדיר את המשתנים הגלובליים IMG_WIDTH, IMG_HEIGHT, BATCH_SIZE.
- מגדיר את המשתנה הגלובלי OUTPUT ומכניס לו ערך של תיקיית הפלט של התוכנית, בה ישמרו כל הקבצים. אם התוכנית רצה דרך Kaggle, התיקייה נמצאת בתוך תיקיית הפלט של הפלטפורמה, אחרת שם את OUTPUT בתיקייה שבה הסקריפט רץ.
- יוצר תיקיות results, models בתוך OUTPUT.
- שואל את המשתמש איזה מסלול ירצה לבחור:
  1. אימון מודל חדש או קיים
  2. חזיית תמונות על בסיס מודל קיים

```
What would you like to do?
1 - model training
2 - predicting color of BW images with existing model
```

- לפי המסלול שנבחר, מזמן את הפעולות הרלוונטיות בסדר הנכון כדי שהתוכנית תעבוד כמו שצריך:
  1. יבוא המידע -> בדיקת המידע -> חלוקת המידע -> יצירת/יבוא מודל -> אימון המודל -> הערכת המודל -> הצגת התוצאות.
  2. יבוא המידע -> בדיקת המידע -> חלוקת המידע -> יבוא מודל -> הצגת התוצאות.
- מדפיס למשתמש הסברים תוך כדי:

```
Due to this deep learning project's purpose, any grayscale image will be excluded.
Any non-image format or image with broken data will also be excluded.
The program works with ONE color format (RGB/RGBA) per run for all the images you choose, based of your first image. Any other
color format will not be accepted.
```

-

מחלקות ופעולות עזר

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

מחלקת Color

```
class Color:
```

- מחלקה שמכילה מחרוזות שמייצגות הדפסה בצבעים ב-Python.

---

myprint

```
def myprint(message, mode, start_newline=True):
```

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

```
def path_verify(message, can_be):
```

מקבלת:

- message – הודעה כמחרוזת.
- can_be – רשימה המכילה מחרוזות של סוג הדרך (path) – קובץ/תיקייה/זיפ (zip).

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

```
def int_verify(massage, min_=1, max_=1000000):
```

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

```
def option_verify(message, options):
```

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

```
def draw_progress_bar(n, total, bar_len=50):
```

מקבלת:

- n – מספר שלם, המיקום הנוכחי באיטרציה.
- total – מספר שלם, גודל האיטרציה.
- bar_len – אורך ההדפסה.

יהונתן טל

מבצעת:

- מדפיסה באורך bar_len סרגל התקדמות של איטרציה, כאשר n/total הוא האחוז הנוכחי.

מחזירה:

- כלום.

```
def iskaggle():
```

iskaggle

מקבלת:

- כלום.

מבצעת:

- מנסה להשתמש בפונקציה שקיימת רק בפלטפורמות שמשתמשות ב-IPython, כמו Jupiter Notebook, ולא כמו cmd וכד'.
- בודקת אם הפלטפורמה היא מבוססת על Jupiter Notebook

מחזירה:

- האם הקוד מורץ בפלטפורמה המבוססת על Jupiter Notebook.

פעולות עזר לבדיקת הקבצים

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

---

find_start_dataset

```
def find_start_dataset(gray=False):
```

מקבלת:

- gray – האם מציאת המאגר ברירת המחדל היא רק לתחזית המבוססת על תמונות שחור-לבן.

מבצעת:

- אם המשתמש רוצה להשתמש במאגר ברירת המחדל, בודקת שהוא קיים בפלטפורמה.
- אם המאגר עדיין בזיפ (zip), מחלצת אותו באמצעות extraction.

מחזירה:

- את הדרך למאגר ברירת מחדל אם קיים בפלטפורמה
- אחרת או אם המשתמש לא רוצה להשתמש במאגר ברירת המחדל, מחרוזת ריקה

---

extraction

```
def extraction(path):
```

מקבלת:

- path – דרך, כמחרוזת.

מבצעת:

- אם הדרך מובילה לזיפ (zip), מחלצת את הזיפ לדרך שתושג באמצעות path_verify.

מחזירה:

- הדרך הנוכחית למאגר המידע, כמחרוזת.

---

scan_dataset

```
def scan_dataset(path):
```

מקבלת:

- path – דרך, כמחרוזת.

מבצעת:

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

יהונתן טל

מחזירה:

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

---

remove_bads

```
def remove_bads(unreadables, wrong_colors):
```

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

```
def 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_PATHS ואם 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_PATHS באמצעות remove_bads.
- חוזרת על התהליך עד שישנו מאגר טוב שמכיל מספיק תמונות טובות.

מחזירה:

- כלום.

---

gray_check_data

```
def 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_PATHS ואם 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_PATHS באמצעות remove_bads.
- חוזרת על התהליך עד שישנו מאגר טוב שמכיל מספיק תמונות טובות.

מחזירה:

- כלום.

---

normal_split_data

```
def normal_split_data():
```

מקבלת:

- כלום.

מבצעת:

- קולטת מהמשתמש את כמות התמונות מהמאגר איתן ירצה לאמן את המודל בעזרת int_verify עם מינימום של BATCH_SIZE*5.
- קולטת מהמשתמש את כמות התמונות מהמאגר איתן ירצה לבחון את המודל בעזרת int_verify עם מקסימום של BATCH_SIZE.
- חוזרת על התהליך עד שסכום הכמויות קטן מסך הדרכים ב-IMAGES_PATHS או שווה לו.
- לפי כמויות אלה, מקצה דרכים אקראיות מ-IMAGES_PATHS ל-TEST_PATHS.
- קולטת מהמשתמש ערך ל-EPOCHS .

מחזירה:

- כלום.

---

gray_split_data

```
def 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_PATHS או 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 הוא מקודד אוטומטי (autoencoder) שמורכב מרשתות עצביות מפותלות (CNN) ומכיל תהליך **קידוד** (encode) ותהליך **פיענוח** (decode) של התמונה: בקידוד השתמשתי בשכבות מפותלות (convolution) עם הפעלת (activation) LeakyReLU. הגדלתי את מספר המסננים (filters/kernels) עם כל שכבה. לשכבות הפענוח, השתמשתי בשכבת UpSampling ואחריה שכבה מפותלת עם מסננים שמספרם יורד בהדרגה, בעצם ההפך מתהליך הקידוד.

הרעיון הבסיסי הוא ששלב הקידוד לוקח קלט ודוחס אותו על ידי העברתו דרך שכבות מפותלות שונות. באמצעות הייצוג הדחוס הזה של הקלט, שלב הפענוח מייצר את הפלט הסופי. מכיוון ששלב הפענוח רואה רק את הייצוג הדחוס של הקלט המקורי, הוא עלול להחמיץ תכונות חשובות של התמונות שאבדו בשלב הקידוד. לכן הכנסתי לכל שכבה בפיענוח את הפלט של השלב המקביל בקידוד, כך שהמפענח יוכל לנצל גם את המידע זה.

---

conv2d

```
def conv2d(layer_input, filters):
```

מקבלת:

- layer_input – השכבה שעליה יתבצע הקידוד – המקודד עד כה.
- filters – מספר המסננים לייצר.

מבצעת:

- מבצעת קידוד באמצעות שכבה מפותלת עם הפעלת LeakyReLU.

מחזירה:

- את המקודד לאחר הוספת השכבות.

---

deconv2d

```
def deconv2d(layer_input, from_encode, filters):
```

מקבלת:

- layer_input – השכבה שעליה יתבצע הפיענוח – המפענח עד כה.
- from_encode – הפלט של השכבה מתהליך הקידוד.
- filters – מספר המסננים לייצר.

מבצעת:

- מגדילה את התמונות עם UpSampling.
- מבצעת פענוח באמצעות שכבה מפותלת.

- מחברת את layer_input ו-from_encode לבאץ' אחד.

מחזירה:

- את המפענח לאחר הוספת השכבות.

---

colorize

```
def colorize():
```

מקבלת:

- כלום.

מבצעת:

- יוצרת מקודד אוטומטי (autoencoder) באמצעות CNN:
- יוצרת מקודד (encoder) שמקבל תמונת Grayscale באמצעות conv2d.
- יוצרת מפענח (decoder) באמצעות deconv2d שמחזיר תמונת RGB.

מחזירה:

- את המקודד האוטומטי כמודל.

---

model_making

```
def model_making(only_load=False):
```

מקבלת:

- 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 colorize ועובר עיבוד.
- אם המשתמש רוצה, מודפס תקציר המודל או נשמרת תמונה ויזואלית של המודל בתיקיית models ב-OUTPUT.
- בסוף הפעולה, MODEL הוא מודל שמקבל תמונת Grayscale ומחזיר תמונת RGB צבועה.

מחזירה:

- כלום.

פעולות אימון (fitting), הערכה (evaluating), ושמירת המודל

פעולות אלה יוצרות את התהליכים הנחוצים כדי לאמן את המודל ולשמור את התוצאות שלו בצורה טובה ואמינה. כמו גם להעריך את המודל באמצעות פונקציות ש-Keras מספקת. בחלק זה קיימת פעולת גנרטור (generator), שלא מחזירה (return), אלא מניבה (yield). כלומר, פעולות מסוג זה מחזירות ערכים בנפרד לאורך זמן – בכל פעם שפונים אליהן הן מניבות ערך אחד מתוך רבים באיטרציה מסוימת.

---

data_transformer_generator

```
def data_transformer_generator(paths, batch_size=BATCH_SIZE):
```

מקבלת:

- batch_size – גודל הבאצ'ים שהגנרטור יעבור עליהם. ברירת מחדל: BATCH_SIZE
- paths – הדרכים לתמונות.

מבצעת:

- מרכיבה כל באץ' מהתמונות שב-path , בהסתמך על תבנית הצבע של התמונות.
- עושה סטנדרטיזציה (standardization) כדי להקל על החישובים במודל ולהפוך אותם ליותר מהירים.

מניבה:

- את הבאץ' כ-RGB בצבע.
- את הבאץ' כ-Grayscale (שחור-לבן).

---

training

```
def training():
```

מקבלת:

- כלום.

מבצעת:

- מאמנת את MODEL עם פעולת fit של Keras בשיטת cross-validation.
- מייצרת את HIST כמילון (Dictionary) שמכיל את המידע של האימון שמתקבל מ-fit.

מחזירה:

- כלום.

---

evaluating

```
def evaluating():
```

מקבלת:

- כלום.

מבצעת:

- מעריכה את MODEL עם evaluate_generator עם כל גנרטורים והערכים המתאימים.
- מדפיסה את נתוני דיוק (accuracy) וההפסד (loss) הסופיים.

מחזירה:

- כלום.

---

model_saving

```
def model_saving():
```

מקבלת:

- כלום.

מבצעת:

- מוחקת את כל המודלים שנשמרו בתהליך האימון חוץ מהאחרון מביניהם, שהוא הטוב מכולם.
- אם המשתמש מעוניין, MODEL נשמר בתיקייה models בתוך OUTPUT. מודל ומשקלים בנפרד.

מחזירה:

- כלום.

---

sampling_graph

```
def sampling_graph():
```

מקבלת:

- כלום.

מבצעת:

- אם המשתמש מעוניין, מציגה גרף של פרטי הלימוד (דיוקים, הפסדים ושיעור הלמידה) לכל חזרה על המאגר (epoch), באמצעות HIST.

מחזירה:

- כלום.

---

sampling_images

```
def sampling_images(grays=False):
```

מקבלת:

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

מבצעת:

- אם המשתמש מעוניין:
- אם grays – מקבלת ערכים מ- data_transformer_generator ומציגה בעזרתם תמונות שחור-לבן, ותמונות עם צבע חזוי לפי המודל, זו לצד זו לשם השוואה.
- אחרת, מציגה גם תמונות בצבע לצד שאר התמונות.
- אם המשתמש מעוניין, התמונות עם הצבע שנחזה נשמרות בתיקייה results ב- OUTPUT.

מחזירה:

- כלום.

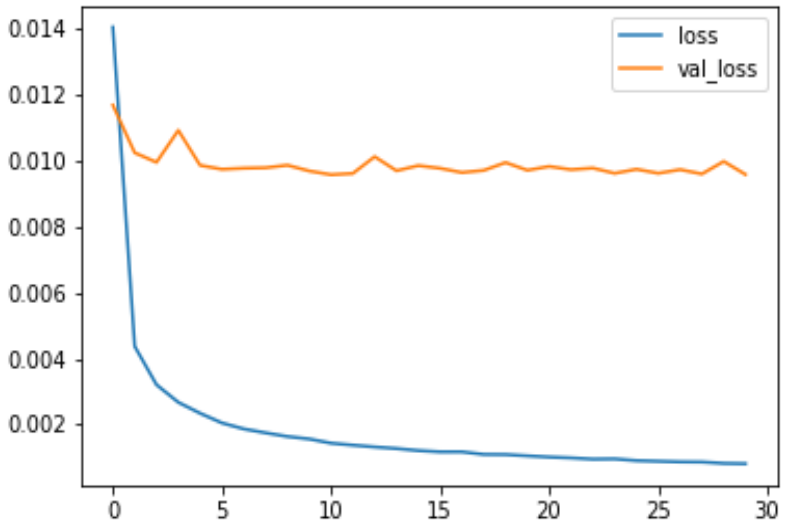
מסקנות הרצת המודל

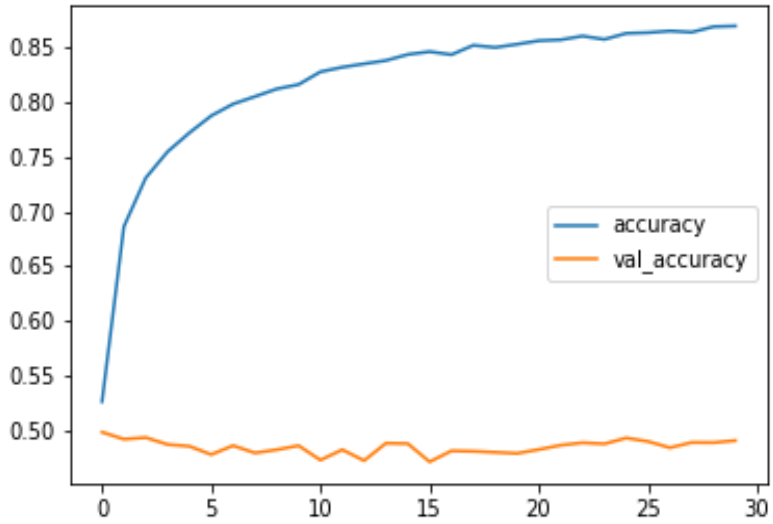
ניסיון ראשון:

המודל רץ על 3000 תמונות ב-30 חזרות (epochs) בזמן כולל של 18 דקות.

loss: 8.1659e-04, accuracy: 0.8694, val_loss:0.0096, val_acc: 0.4909

אפשר להבין מהתמונות בעמוד הבא ומהגרפים שלא הייתה התאמה מספרית לכל אורך האימון בין התוצאות על תמונות האימון לבין התוצאות על תמונות הוולידציה. לאחר מחקר באינטרנט הבנתי שהסיבה לכך היא התאמת יתר (overfitting) לתמונות האימון. אז הוספתי שכבות Dropout, הוספתי פעולות רגולציה (regularization), והגדלתי את כמות התמונות. אלה אמורים לעזור במקרה של התאמת יתר, אך דבר לא עזר. חשוב לציין שבמקרה זה התוצאה לא הייתה כל כך נוראית משום שהיא כן סיפקה רמה של צבע אמין, גם אם לא נכון.





צביעת תמונות שחור-לבן

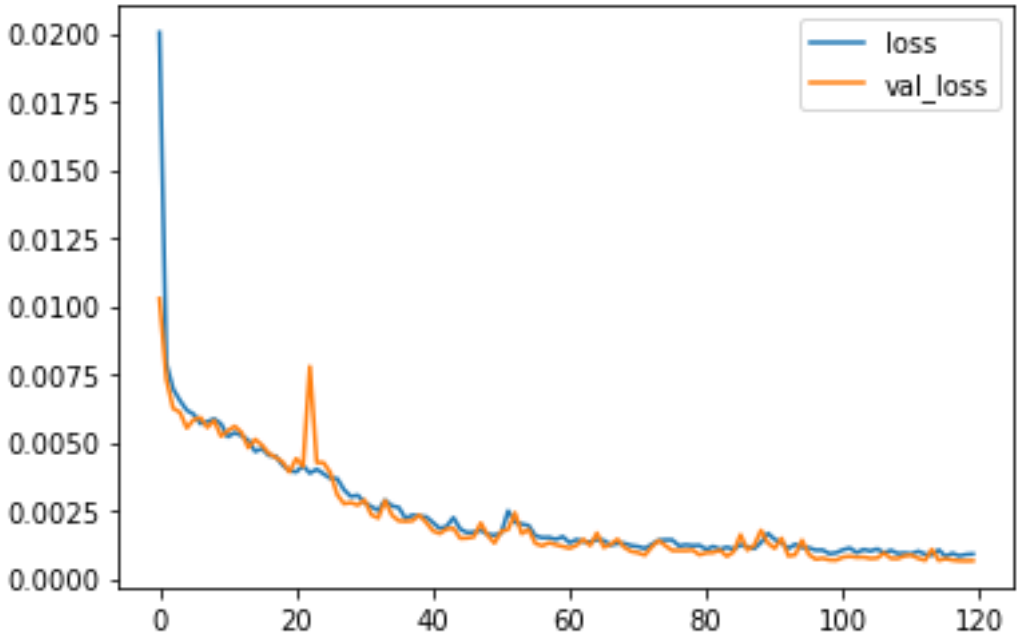
יהונתן טל

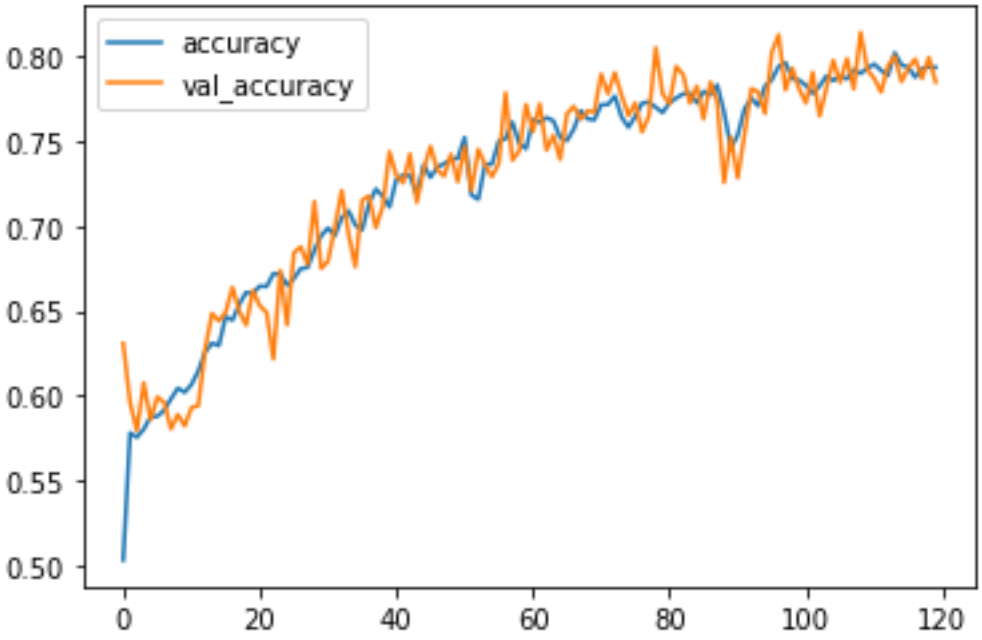
ניסיון שני:

המודל רץ על 3000 תמונות ב-120 חזרות בזמן כולל של 74 דקות.

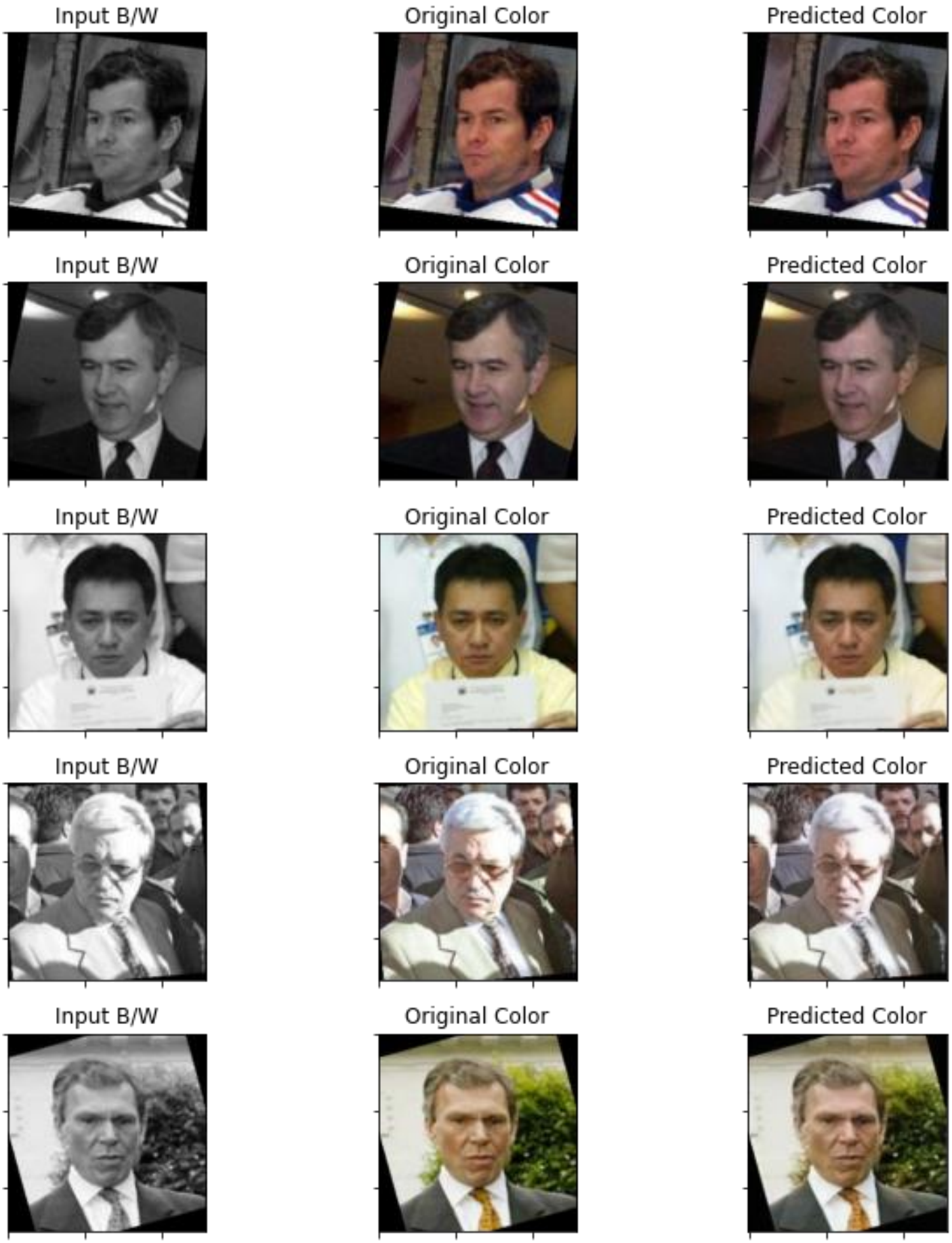
loss: 9.0687e-04, accuracy: 0.7938, val_loss: 6.6080e-04, val_accuracy: 0.7851

הפעם השתמשתי בשיטת cross-validation כדי להימנע מהתאמת יתר באמצעות שינוי התמונות שמשמשות לאימון ולווידוא, כל פעם שהאימון חזר על המאגר (epoch). התוצאות נראות בגרפים ובתמונות, אך היה צורך במספר רב יחסית של חזרות כדי להגיע לנתונים האלה.





צביעת תמונות שחור-לבן



סיכום אישי

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 ופעולות שמצאתי באינטרנט בניסיונות עוורים אולי להצליח במשימה שלי. לאחר חודש או שניים בעת ניסויי היותר-מוצלחים עם קוד תכנית 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 הוסברו הרבה נושאים ופקודות 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 לייאוש גמור 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 ב-Python, 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.

לאחר סיום כתיבת הקוד וחצי מהספר, התמודדתי עם הבעיה האחרונה שלקחה לי הרבה מחשבה כדי לפתור – התאמת יתר. כפי שפורט בחלק על מסקנות המודל, הגעתי לתוצאות הסופיות לאחר מעבר בדרך חתחתים לא קצרה. הרעיון לפתרון הסופי הגיע כמובן מהאינטרנט, אך הביצוע היה מאתגר. לבסוף הגעתי לתוצאות מספקות, אפשר לראות שהמודל באמת יודע לעשות את העבודה שלו (:

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 ועם הפנאי 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.

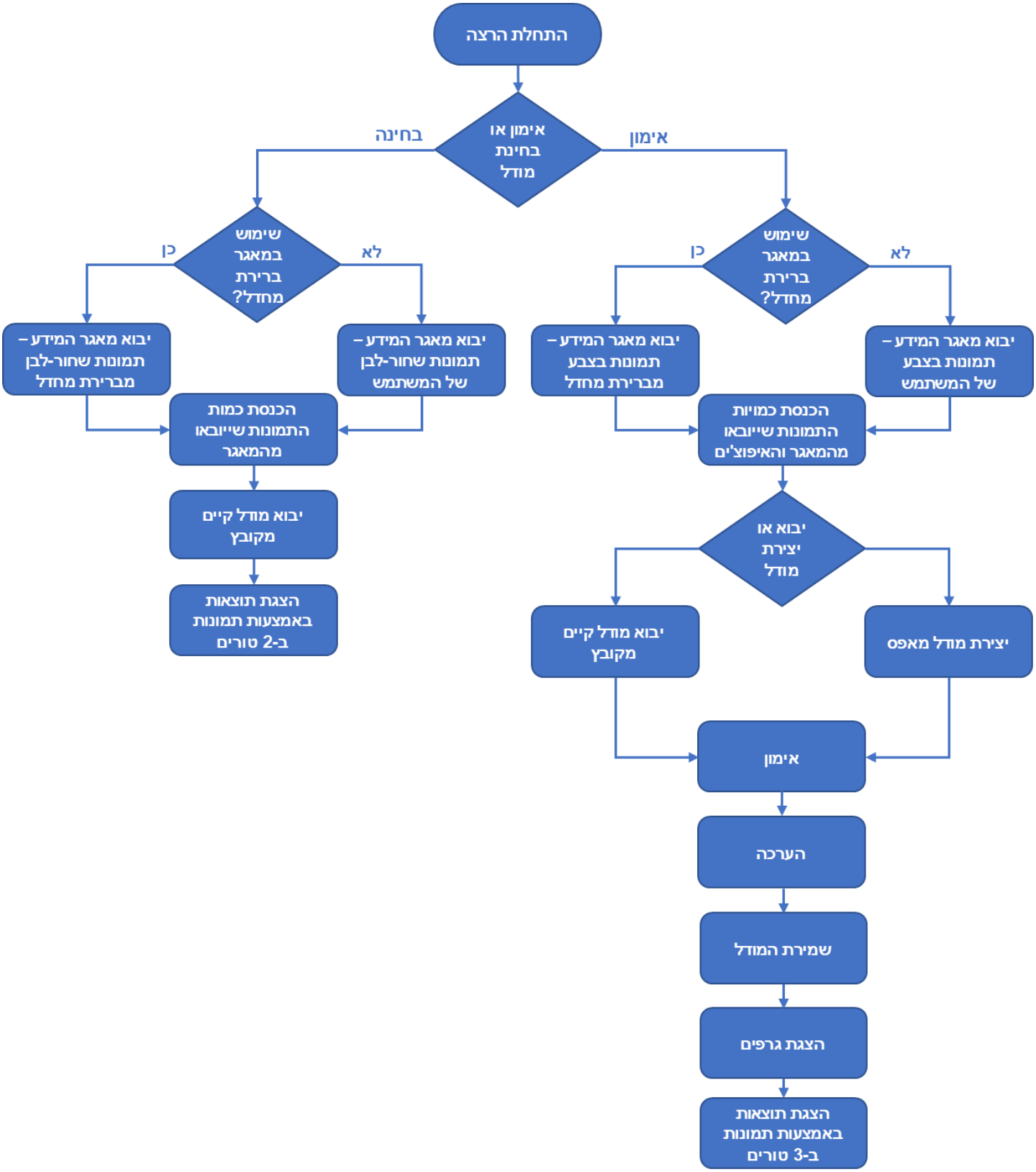
ביבליוגרפיה

- https://www.kaggle.com/
- https://keras.io/api/applications/
- https://jupyter.org/
- https://github.com/BryanPlummer/flickr30k_entities
- https://fairyonice.github.io/Color-gray-scale-images-and-manga-using-deep-learning.html
- https://towardsdatascience.com/reduce-memory-usage-and-make-your-python-code-faster-using-generators-bd79dbfeb4c
- https://sanjayasubedi.com.np/deeplearning/black-and-white-to-color-using-deep-learning/
- https://becominghuman.ai/auto-colorization-of-black-and-white-images-using-machine-learning-auto-encoders-technique-a213b47f7339
- https://blog.floydhub.com/colorizing-b-w-photos-with-neural-networks/
- https://keras.io/guides/functional_api/
- https://www.pyimagesearch.com/2018/12/31/keras-conv2d-and-convolutional-layers/
- https://www.pyimagesearch.com/2016/07/25/convolutions-with-opencv-and-python/
- https://developers.arcgis.com/python/guide/how-unet-works/

נספחים

תרשים מסלול הרצה

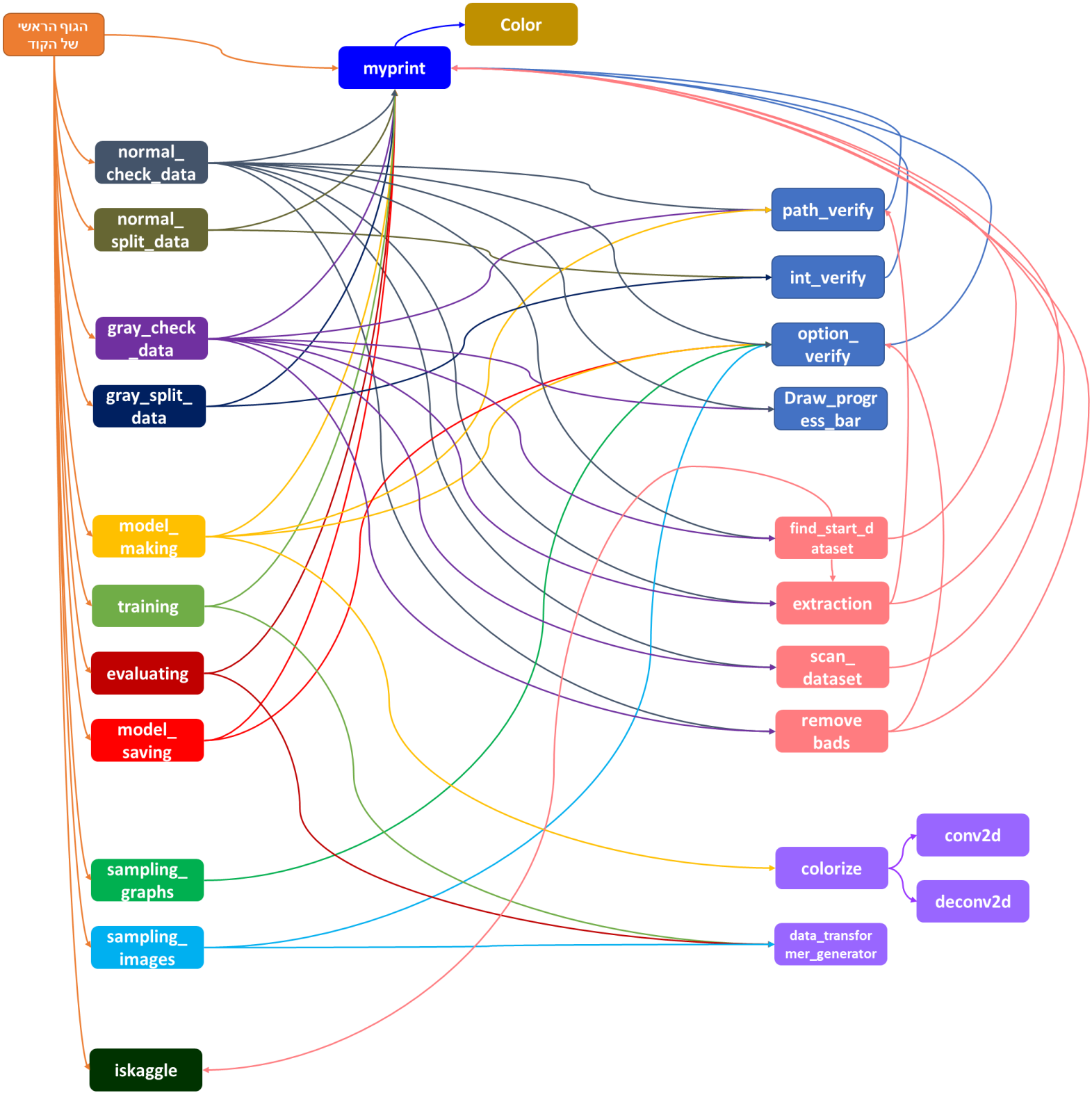
זהו תרשים המתאר את מסלול ההרצה הכללי של התוכנית **מבחינת המשתמש**.



יהונתן טל

תרשים קריאות

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



ייצוג ויזואלי של המודל

התמונה ש-Keras מפיק גדולה מדי למסמך אז היא זמינה בקישור:

https://1drv.ms/u/s!AsNZggxhuzRzktk9Gc1pfs2v7kEPyg?e=AYO78I

יש גם ייצוג תלת-מימדי מ-http://alexlenail.me/NN-SVG/AlexNet.html שמציג בצורה די טובה את 27 שינויי הגודל שעוברת תמונה במודל. כל תיבה מייצגת מערך ארבע-מימדי (טנסור tensor / באץ') שמועבר למודל, כשהגודל והרוחב מייצגים את עצמם ומספר הערוצים הוא העומק של התיבה. החיצים הכחולים-אדומים מייצגים את גודל ה-kernels, קטעי תמונה קטנים עליהם המודל מבצע חישובים. שינויי הגודל מתרחשים כחלק מתהליך הקידוד האוטומטי ומתבצעים באמצעות השכבות conv2d, Concatenate, UpSampling2d.