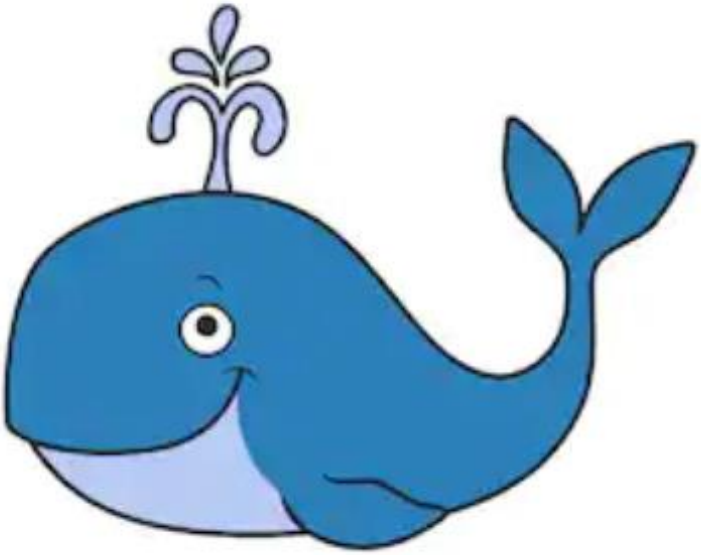
Humpback Whale Identification



תוכן עניינים

Abstract

במסגרת הקורס בחרתי לבצע פרויקט של זיהוי לוותינים שהוא חלק מתחרות ב-"קאגל".

הליך העבודה כולל ארבעה מרכיבים:

- בחירת סביבת עבודה
- ארגון ועיבוד מקדים של הנתונים.
- פירוט המודל בו השתמשתי בתהליך העבודה.
- חיזוי תמונות הבדיקה באמצעות המודל

מצורפים הקבצים:

- project.ipynb – מחברת jupyter, שבה פותח הקוד.
- Run_model.py- סקריפט שמכיל את הקוד מהמחברת, קל להרצה.
- Submission.csv- קובץ ההגשה שהמודל יצר.
- תיקיית Matlab- עם קוד המתלב וה-bounding boxes שהוא יצר לtest/train בתור קובץ אקסל.

תיאור קוד הפרויקט

*המושג "חלון" – קטע קוד במחברת jupyter הנמצא בקטע Code אחד (נראה כמו חלון).

בקוד המצורף (מחברת Project.ipynb) לכל חלון ניתנה כותרת. לדוגמא, החלון השני מלמעלה כותרתו היא – "Constants". לכל חלק במסמך יש הפנייה לחלון המתאים, כדי להקל על מציאתו, בנוסף לתיעוד, נרשם הסבר קצר על הנעשה בכל חלון, ההסבר חוזר על עיקרי הדברים הרשומים במסמך זה.

הפרויקט נבנה באופן מודולרי כך שכל חלון יכול לרוץ עצמאי ללא קשר לחלונות האחרים. בתחילת כל חלון מבוצעת הכנה של הנתונים הדרושים לו ובסופו מתקבל פלט (נדרש להריץ פעם אחת בתחילת הקוד את חלון ה import וה constants), כך שניתן לבחון כל חלון באופן עצמאי.

כמו כן , הקוד חולק ל- 3 חלקים עיקריים:

- Preprocessing
- Training
- Predictions

ניתן לדלג על אחד משני השלבים הראשונים או על שניהם ביחד ולהשתמש בתוצאות שהתקבלו מההרצות הקודמות של כל חלק, באמצעות שינוי האינדיקציות הרשומות בחלון "Constants":

- START_FROM_SCRATCH – להתחיל את כל התהליך מההתחלה (ברירת מחדל: False).
- DO_TRAINING- ערך True, מדלג על הPreprocessing. ערך False, מבצע רק את הPredications (ברירת מחדל: True).

כיוון שבמהלך העבודה חוויתי ניתוקים מהרשת והצורך להריץ את תהליך ה-Training הממושך, העברתי את הקוד למסמך פייתון (run_model.py) והרצתי אותו על השרת המרוחק.

```
##############################################  Constants ##################################################################

SAMPLE_FILE           = '/home/or/whales/sample_submission.csv' # The sample submission file
TEST_FOLDER           = '/home/or/whales/test'                  # Folder of all the test pic
TRAINING_FILE         = "./train.csv"                           # The test csv file
TRAIN_FOLDER          = '/home/or/whales/train'                 # Folder of all the train pic
H2PS                  = "/home/or/whales/h2ps.pickle"           # All the oic tha has the same hash
P2H                   = "/home/or/whales/p2h.pickle"            # Pic to hash matching
H2P                   = "/home/or/whales/h2p.pickle"            # Hash to pic matching
P2SIZE                = "/home/or/whales/p2size.pickle"         # Size of picures
CONV_TRAIN_FILE       = "/home/or/whales/conv_train.csv"        # Train.csv after update close pictures
CONV_SUB_FILE         = "/home/or/whales/conv_sub.csv"          # Submission.csv after update close pictures
TRAINING_FILE_3_NO_W  = "/home/or/whales/train_3channel.csv"    # Updated train.csv after transform all pic to 3 cahnnels wit
TRAIN_FOLDER_3CHANNEL = '/home/or/whales/train_3channel'        # Updated train.csv pic directory
TRAINING_FILE_FINAL_A = "/home/or/whales/train_aug_final.csv"   # Updated train.csv after transform with augmentation of pict
TRAINING_FILE_FINAL   = "/home/or/whales/train_final.csv"       # Train file without the validation set
TRAIN_FOLDER_FINAL    = '/home/or/whales/train_final'           # Directory of updated train.csv after transform with augment
VALID_FILE_FINAL      = "/home/or/whales/validation_final.csv"  # Validation file the final csv file
VALID_FOLDER_FINAL    = '/home/or/whales/validation_final'      # Validation directory
BB_TRAIN              = '/home/or/whales/BB_train.csv'          # Boundin boxes file for training
BB_TEST               = '/home/or/whales/BB_test.csv'           # Boundin boxes file for test
SUBMISSION_FILE       = '/home/or/whales/submission.csv'        # Submission file
HISTORY_FILE          = '/home/or/whales/history.pkl'           # The history of the model
MODEL_TIME            = '/home/or/whales/Model_tictoc.pkl'      # The runung time for the model
BATCH_SIZE            = 100                                     # Batch size
IMAGE_SIZE            = 224                                     # Image one dimention
TARGET_SIZES          = (IMAGE_SIZE,IMAGE_SIZE)                 # Total image dimentions
CLASSES               = 5004                                    # Number of classes
EPOCHS                = 120                                     # Number of epochs
VALIDATION_STEPS      = 100                                     # Number of steps in the validation
NEW_WHALE_INDEX       = 0                                       # Index of the nw whale for the test time augmentation
NUMBER_OF_PREDICATIONS= 5                                       # Number of predications per whales, for the test time augmer
THRESHOLD             = 0.276                                   # Threshold of certainty for the test time augmentation
MODEL_FILE            = "/home/or/whales/Or_model.h5"           # Name if the the previous trained model
VALIDIDATION_PRECENT  = 0.3                                     # Number of pictures taken to the validation
EPOCH_WAIT_IMPROVE    = 12                                      # After this number of epochs, if thre is no improvment in th

##############################################  Indications ################################################################
START_FROM_SCRATCH    = False                                   # Indication if to do all the preprocessing all over again(or
DO_TRAINING           = True                                    # Indication if to train the model again or use the existing
```

איור(1)- חלון קוד ה"Constants" במחברת jupyter.
האינדיקציות נמצאות בתחתית החלון.

סביבת העבודה

בחירת סביבת העבודה – במהלך העבודה, ניסיתי מספר סביבות פיתוח, ונמצא כי כל סביבה עבודה נותנת אפשרויות שונות.

בחירת סביבת העבודה(השתנתה עם הזמן):

1. Google collab- סביבה עבודה שהשתמשנו במהלך קורס, היא מאופיינת בזכרון אחסון של 15GB בלבד. ניסיתי להשתמש במקביל במספר חשבונות ולהעביר נתונים מאחד לשני. לאחר מספר ניסיונות ,עלה כי התהליך מאוד איטי ומסורבל, לכן בחרתי לא להשתמש בסביבת עבודה זו .

2. Kaggle- סביבת עבודה במסגרת התחרות, בעלת זמן עיבוד מהיר יותר מ-Google collab. בסביבת עבודה זו לא ניתן לשמור קבצים (הם נמחקים אחרי timeout של כ- 6 שעות) ומאחר וקיימת הנחייה לפיה אסור להשתמש בחומרים אחרים ברשת, הביא אותי לעזיבה של סביבת עבודה זו.

3. Google Cloud- מכונה וירטואלית בגוגל בה ניתן להגדיר זיכרון ואף לשמור קבצים.
סביבת עבודה זו הינה בתשלום.

בסופו של דבר נבחרה סביבת Google Cloud, הסביבה הוקמה עם 100GB של אחסון ועם GPU אחד של NVIDIA Tesla P100, כאשר זיכרון הRAM של המחשב הוא 30GB.

עיבוד הנתונים

סט התמונות הכלול בפרויקט הינו מאתגר במיוחד, מהסיבות הבאות:

1. אחוז גבוה מתמונות הושמט מהליך האימון; 9664 תמונות מתוך 25361 תמונות אימון, (~38%) הם 'new_whale", שפירושו שקול ל- "אין לנו מושג איזה לוויתן זה", (ראה TTA בהמשך).
2. מקבץ תמונות של gray scale יחד עם תמונות color, בגדלים שונים(הכי גדולה היא 1050X750).
3. תמונות רבות מטושטשות.
4. סיווגים רבים (2073 סיווגים, כ~41%) בעלי תמונה אחת בלבד.
5. תמונות כפולות. תמונה שהופיעה גם בסט האימון וגם בסט הבדיקה, בנוסף עוד תמונה שהופיעה פעמיים בסט האימון.

<u>סדר בסט התמונות (בקוד: Finding similar images and update the records):</u>

לפי [1], סט הנתונים ששימש בתחרות קודמת הכיל תמונות כמעט זהות עם סיווגים שונים. בהסתמך על עבודה זו השתמשתי באותו רעיון, לזהות תמונות כמעט זהות.
זיהוי תמונות מאוד קרובות אחת לשנייה מבוסס על יצירת פונקציית גיבוב לכל תמונה ובדיקה של מרחק פונקצייה כזו לפונקציית תמונה אחרת. פונקציית הגיבוב שנבחרה היא של [2] אשר משמשת לזהות תמונות דומות בתעשייה כדי למנוע copy rights violation.

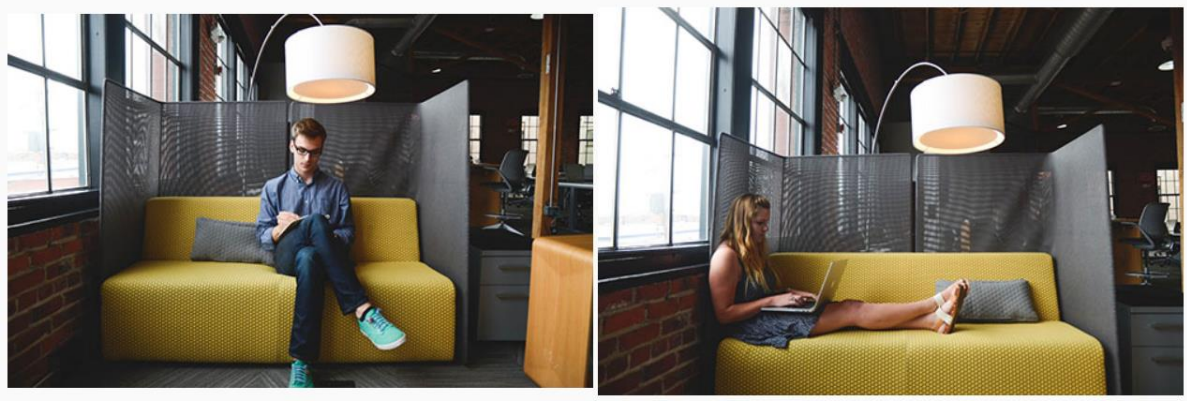
תמונות קרובות נבחרו עם דמיון של לפחות 90% אחוזים בניהן ושמרחק ה-Hamming יהיה קטן מ-6.

קרבת דמיון:

איור(2)- אילוסטרציה של קרבת דמיון, לפי[3], התמונה משמאל(המקורית) עם יחס קרבת דמיון של 96.8% לתמונה האמצעית(אחרי grayscale) ויחס קרבת דמיון של 78.1% לתמונה הימנית(אחרי cropped, increased color saturation).

מרחק Hamming:

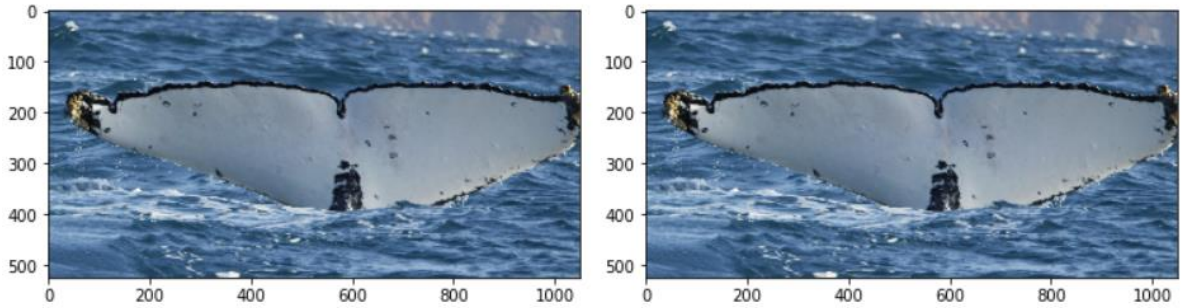
איור(3)- אילוסטרציה של מרחק Hamming, לפי [4], ל-2 התמונות יש מרחק של 3, ניתן לראות שהן זהות.



איור(4)- אילוסטרציה של מרחק Hamming, עם מרחק של 32, ניתן לראות שהתמונות מאוד דומות.

במהלך הבדיקה (Finding similar images and update the records:Part 2), נמצאה רשומה זהה של אותו לוויתן (אותה תמונה), הנמצאת בtrain וגם בtest, לכן הוענק לה Id מה- train (סיווג של new_whale).

כמו כן נמצאו 2 תמונות מאוד קרובות אחת לשנייה. אם נבחן אותן לעומק, נמצא שהן ממש זהות:



איור(5)- שתי התמונות הזהות הנמצאות בסט האימון.

כיוון שהתמונות מאוד קרובות אחת לשנייה ובעלות אותו הסיווג(w_7a8ce16), נמחקה אחת מהן.

פלט של חלון: הקובץ עם הרשומות המעודכנות של נתוני האימון נשמר לפי הקבוע: CONV_TRAIN_FILE ושל נתוני הבדיקה נשמר לפי הקבוע: CONV_SUB_FILE.

<u>יישור התמונות לפי אותו מימד וסינון סיווג(בקוד: Formatting the images):</u>

במסד הנתונים ישנן תמונות צבעוניות ותמונות אפורות. על מנת להפוך את כל התמונות לייצוג זהה, וללא הסתמכות על המודל שיעשה זאת עבורנו, כל תמונה הוסבה ל-3 ערוצים , במצב בו תמונות רק בעלות ערוץ אחד, הן הוכפלו 3 פעמים.

בנוסף, מאחר שסיווג של לוויתן חדש (new whale) משמעותה אי ידיעת סוג הלוויתן, הושמטו כל הסיווגים של הלווייתנים החדשים (ראה TTA בתהליך החיזוי בהמשך).

פלט של חלון: בסוף התהליך נשמרו התמונות בתיקיה הרשומה בקבוע: TRAIN_FOLDER_3CHANNEL. והקובץ עם הרשומות המעודכנות של נתוני האימון נשמר לפי הקבוע: TRAINING_FILE_3_NO_W.

יצירת אוגמנטציות ויישום של Bounding Boxes(בקוד Create augmentations):

מאחר ובהליך זיהוי תמונות רק הזנב של הלוייתן חשוב ולא הרקע, ייצרתי bounding boxes שיתחמו את הזנב על מנת לקבל תוצאות טובות יותר בהרצת המודל.

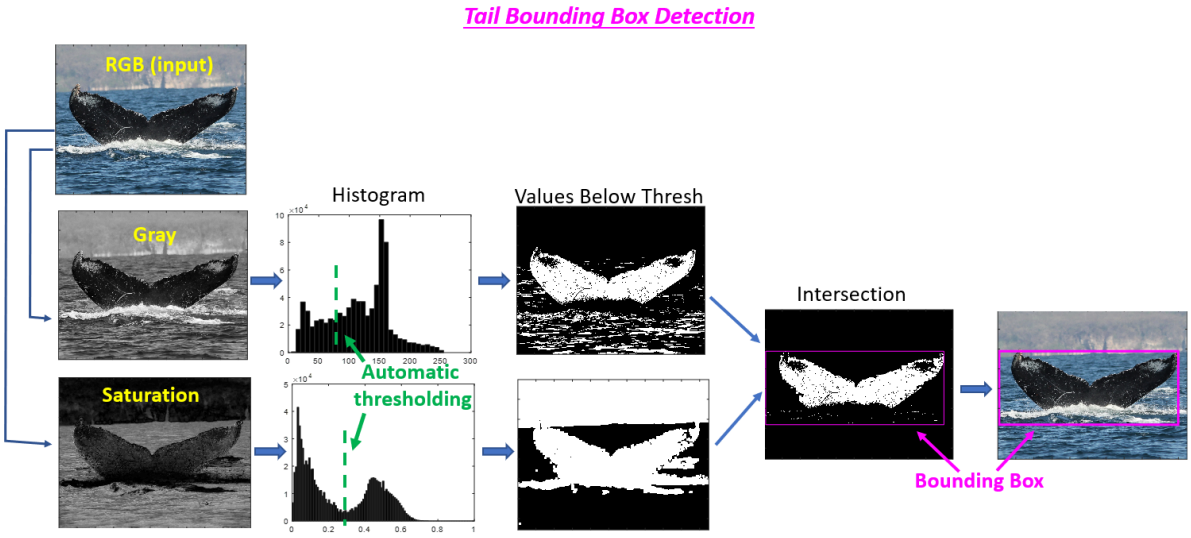
ה bounding boxes יוצרו ב- MATLAB(ראה קוד מצורף), והושמו על פי הלוגיקה כמפורט:

- כל תמונה הומרה לHSV, ממנה לקחו הsaturation וההמרה של הV ל-Gray scale.

- בחלקי התמונות של הזנבות זוהו שני דפוסים;

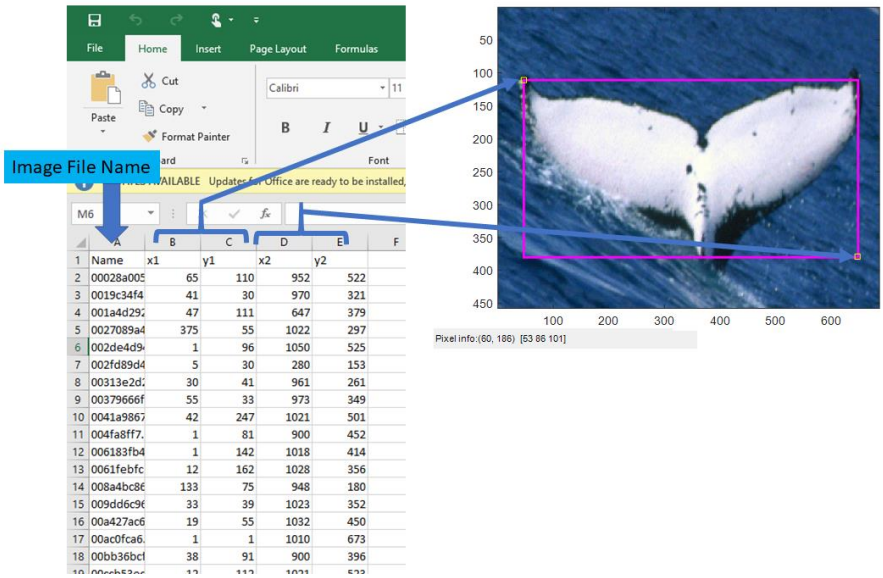
1. כאלה שיש להן saturation ועוצמת ו-Gray scale נמוכים מהרקע:
   - נמצאו שני thresholds בהיסטוגרמה של התמונות ונלקחו הsaturation וה-Gray scale הנמוכים מהסף התחתון.
   - בוצע חיתוך בין שתי התמונות שהתקבלו ונלקחו 2 קואורדינטות שתוחמות את הפיקסלים שנמצאו באיחוד.
2. כאלה שיש להן saturation גבוהה ועוצמת ו-Gray scale נמוכה מהרקע:
   - נמצאו שני thresholds בהיסטוגרמה של התמונות ונלקחה הGray scale הנמוך מהסף התחתון וה-saturation הגבוה מהסף העליון.
   - בוצע חיתוך בין שתי התמונות שהתקבלו ונלקחו 2 קואורדינטות שתוחמות את הפיקסלים שנמצאו באיחוד.

- זנבות שהBoundig boxes שהתקבלו היו קטנים מ10%, נחשבו כלא מוצלחים.



איור(6): איור סכמתי של לוגיקת יצירת הBounding boxes.

הקבצים נשמרו בתור csv ומיקומם נמצא בקבועים BB_TRAIN ו- BB_TEST, כמתואר באיור (7)



איור(7): קובץ הbounding boxes ומיקומי הנקודות היוצרות ריבוע התוחם את הזנב.

בסוף התהליך ל-16672 תמונות יוצרו bounding boxes . כל שאר התמונות פשוט הותאמו לגודל התמונה שהמודל קיבל(ראה חלון Applying the augmentations בקוד).

דוגמאות:

איור(8): דוגמאות לbounding boxes התוחמות את את הזנב.

לאחר יצירת מסד נתונים של התמונות , לכל תמונה בוצעו ארבעה סוגי אוגמנטציות (בחלון Defining the augmentations בקוד):

- הפיכת התמונה שמאלה-ימינה והוספת טשטוש גאוסי.
- הזזת התמונה רנדומלית בטווחים: שמאלה-ימינה(0-20 פיקסלים) ולמעלה-למטה(0-20 פיקסלים) ושינוי התמונה לגודל הקלט של המודל(לפי הקבוע IMAGE_SIZE).
- הזזת התמונה באותו קנה מידה כמו באוגמנטציה הקודמת והפיכת התמונה שמאלה-ימינה.

התמונה המקורית:

איור(9): תמונה מקורית בסט האימון.

האוגמנטציות שלה:

איור(10): האוגמנטציות של התמונה מקורית בסט האימון.

בסוף התהליך נשמרו התמונות לתיקיה הרשומה בקבוע: TRAIN_FOLDER_FINAL.
והקובץ עם הרשומות המעודכנות של נתוני האימון נשמר לפי הקבוע: TRAINING_FILE_FINAL.

המודל

המודל (בקוד Training):

נעשה שימוש במודל המבוסס על החומר שנלמד בקורס, InceptionResNetV2.
המודל השתמש בכל השכבות של InceptionResNetV2, ללא ה-Fully connected והוספו עליו average pooling בגודל של 2X2, Drop out של 40% ו-Fully connected אחד שמחשב את הLoss בעזרת Softmax. בנוסף לפי קורס c231n של סטנפורד[5], השתמשתי באופטימייזר המומלץ של Adam.

בחירת מודל עם Fully connected נמוך, כיוון שאימון המודל דורש זמן רב. כמו כן השתמשתי בכל השכבות המאומנות של המודל InceptionResNetV2(מתחת לFully connected), כפשרה בין זמן ריצה למידת דיוק טוב יותר(כמו שראינו בתרגיל הקודם בקורס).

כדי לקבל תחזיות יותר טובות ולהימנע מ-overfitting, יצרתי validation set אשר מהווה 30% מסך כל התמונות ונמצאות בו לפחות תמונה אחת מכל class (יצירת סט "ולידציה" נמצאת בקוד ב:Create validation set . הערה: על מנת לחסוך במקום, על חשבון מודולריות מלאה, הועברו התמונות מהסט הכולל וכך נוצרו סט האימון וסט ולידציה. כדי להריץ את החלון הזה יש צורך להריץ גם את החלון שלפניו ,שיוצר את הסט הכולל לפני הפיצול לשני הסטים).

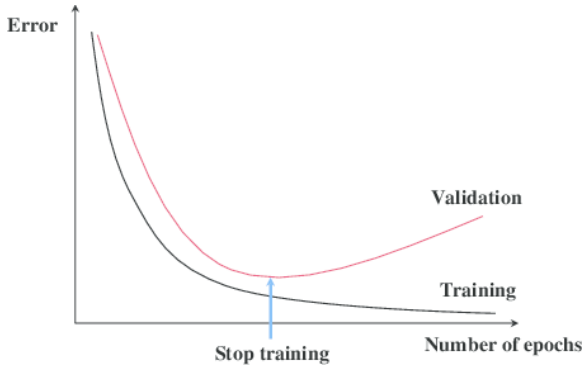
הקלט למודל היה:

- סט אימון של 43299 תמונות מ5004 סיווגים שונים
- סט ולידציה של 19485 תמונות מ5004 סיווגים שונים

על מנת לקבל מגוון רחב יותר של תוצאות השתמשתי גם באוגמנטציות של תמונות בזמן אימון המובנה של keras.

לקבלת מודל הטוב ביותר ולהימנע מoverfitting נוצר תנאי עצירה: אם הvalidation accuracy, לא משתפר תוך 5 Epochs, אז תהליך האימון עוצר והמודל נשמר, ראה איור 11.

לא ביצעתי שמירת המודל תוך כדי אימון, אחרי כל Epoch של שיפור הvalidation , מאחר שנדרש זמן ארוך לשמירה בכל פעם (בערך 6 דקות לכל שמירה) והסתפקתי במודל הכי טוב בטווח של 5 Epochs.



איור(11)- overfitting של המודל ונקודה בה המודל המאומן הוא הטוב ביותר.

בוצעו מס' רב של הרצות ובכל אחת מההרצות המודל הגיע לoverfitting . בכל הרצה שונו ההיפר-פרמטרים:

- נבדקו האופטימייזרים המומלצים הבאים לפי cs231n, [5]: Nesterov Momentum ו-Adam ושונו הערכים שלהם, עד שבסוף נקבע הערכים המומלצים לפי[7].
- שונו ערכי הdropout.
- הוספו ערכי L2/L1/L1_L2 regularizations.
- הוספו Fully connected layers נוספים עם מספר שונה של nodes.
- נלקחו משקלים מאומנים ממספר שכבות משתנה של המודל.
- נבדקו המודלים InceptionResNetV2 ו- ResNet50.
- הורץ המודל עם הסיווג new_whale.
- עם מספר רב יותר של אוגמנטציות(4 במקום 3, יצר סט אימון של 54936 וסט ולידציה של 23544).
- בלי early stopping.
- שונים Batch sizes.

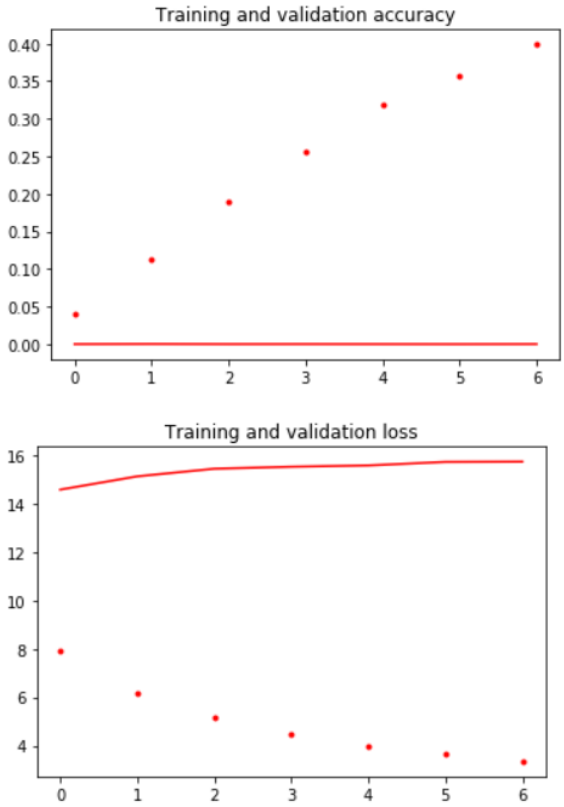
בכל ההרצות קיבלתי את אותה התוצאה:

Name	Submitted	Wait time	Execution time	Score
submisssion.csv	just now	0 seconds	0 seconds	0.276
Complete				

איור(12)- התוצאה מkaggle.

המודל רץ במשך 5085.564310 שניות.

*מצורפים מספר צילומי מסך של ריצות לדוגמה עם היפר-פרמטרים שונים בתקייה Previous run example , (ברוב הריצות הפלט במסך היה של כל איטרציה של batch ולא כל epoch ולכן לא היה אפשרי להראות את התוצאה בצורה קומפקטית ע"י צילום מסך אחד).

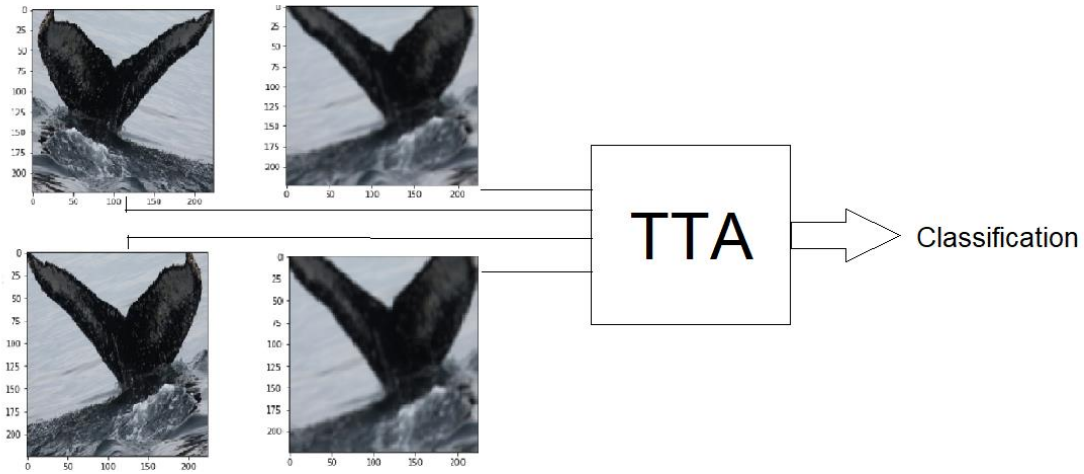


איור(13)- גרפים של הדיוק והloss עבור סט הולידציה וסט האימון.
סט האימון מסומן בנקודות וסט הולידציה בקו רציף.
ניתן לראות בצורה ברורה שמגיעים לoverfitting, כיוון שהloss של סט האימון יורד ושל סט הולידציה עולה.

חיזוי(בקוד Predictions):

על מנת לחזות את הסיווג של תמונות הtest, השתמשתי בTest Time Augmentation(TTA), לפי מקור [6], שיטה זו נתנה שיפור בתוצאות ואפשרה להכניס את הסיווג "new whale" שהושמט מתהליך האימון.

הרעיון מאחורי TTA, הוא שהחיזויים אמורים להיות כולם או רובם אותו הדבר, גם בשינויים של תמונה(מבצעים עליה אוגמנטציה). כמו כן, אם קיים ביטחון גבוה לגבי הסיווג הנכון, ניתן להוסיף סיווג אחר, במקרה שלנו "new whale".



איור(14) – אילוסטרציה לקלט לTTA והפלט שלו.

ארבעה גרסאות של חיזוי תמונות הוכנסו לTTA:

1. התמונה מקורית.
2. היפוך התמונה המקורית בכיוון שמאל-ימין.
3. הזזת התמונה רנדומלית בטווחים: שמאלה-ימינה(0-20 פיקסלים) ולמעלה-למטה(0-20 פיקסלים) ושינוי התמונה לגודל הקלט של המודל(לפי הקבוע IMAGE_SIZE).
4. הזזת התמונה באותו קנה מידה כמו באוגמנטציה הקודמת והפיכת התמונה שמאלה-ימינה והוספת טשטוש.

<u>החשיבה בשימוש TTA:</u>

עבור כל תמונה שלא קבלה חיזוי מראש(שנמצאת גם בסט האימון – בוצע בPreprocess)מבוצעות הפעולות כמפורט:

- במצב בו לתמונה קיים Bounding box, או לא קיים Bounding box יתבצע resize לגודל התמונה שמתאים למודל.
- בנוסף לתמונה המקורית מייצרים עוד שלושה אוגמנטציות.
- כל ארבעת התמונות יעברו חיזוי ויבוצע עבור כל אחת מהן חמישה חיזויים וניתן אינדקס לכל אחד מהם המבטא את מידת החיזוי [מהחיזוי הטוב ביותר לנמוך].
- החיזויים הטובים ביותר הוכנסו לרשימה הסופית לפי הסדר(מסיווג גבוהה לנמוך) באופן הבא:
  - נבדק הסיווג הכי נפוץ בכל אחד מכל המיקומים של האינדקסים של החיזויים, אם יותר מחיזוי אחד מצביע על אותו סיווג , הוא יכנס לרשימה הפלט הסופית (החיזוי הבא הכי סביר לאותה גרסה של התמונה), אחרת:
  - נלקח הסיווג בעל סבירות הכי גבוהה מכל המיקומים של האינדקסים של החיזויים לתוך רשימת הפלט הסופית (החיזוי הבא הכי סביר לאותה גרסה של התמונה).
  - העברת האינדקס של החיזוי הנבחר לסיווג הבא עם החיזוי הכי גבוהה.
  - סיווג שלא עבר את הסף(Threshold = 0.276) יסומן במקום לפניו הסיווג new whale".

סיכום

בוצעו ניסיונות רבים על מנת לגרום למודל להימנע מoverffitingולהגיע לתוצאות טובות יותר. כל שלב בקוד (אוגמטציה, חלוקה לסט training ו- validation וכו'), נבדק מספר פעמים לוודא שפועל כשורה. המסקנה שהתקבלה הצביעה על כך שההיפר-פרמטרים של המודל אינם מכוונים נכון. לפי קורס cs231n של סטנדפורד, קביעת ההיפר-פרמטרים דורשת חיפוש רחב טווח במרחב הפרמטרים האפשריים ונדרשות הרצות רבות עד למציאת הערכים האופטימליים. לכן אני סבור שהמודל הגיע למינימום מקומי ונדרשות עוד בדיקות עם היפר-פרמטרים שונים.

ביבליוגרפיה

[1]https://www.kaggle.com/seesee/siamese-pretrained-0-822
[2] https://www.phash.org
[3]https://cloudinary.com/blog/how_to_automatically_identify_similar_images_using_phash
[4]https://packagist.org/packages/jenssegers/imagehash
[5]http://cs231n.github.io/neural-networks-3/
[6]https://www.kaggle.com/andrewkh/test-time-augmentation-tta-worth-it
[7]https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/