למידה עמוקה בפייתון
פרויקט שלב ב'

Fire-Detection Using TensorFlow
Pretrained SSD MOBILENET v2 FPNLITE 640x640
Object detector model

מה עושה התוכנית

התוכנה מקבלת מסד נתונים מאורגן, מודל נבחר עם הערכים המתאימים(ערכי הייפר) ומשם מאומן לזהות שריפות\אש, אחרי אימון נעבור לשלבי הרצתה וחיזוי של המודל על מסדי הנתונים המוגדרים כקבוצת בחינה למודל. התוכנה אמורה לגלות שריפות ולהתריע על כך ובכך תעזור למנוע שריפות שעלולות להתפשט, אנו בוחנים אופנים שונים בהם אפשר לתת למודל לגלות שריפות.

קלט

בסופו של דבר, הקלט של התוכנית יכולה לבוא בכמה דרכים, ווידאו אם זה הקלטה או שידור חי ותמונות.

פלט

הפלט תלוי בקלט שאנו עובדים איתו, אם זו ווידאו של הקלטה כלשהי, הפלט יכול להיות קובץ של ההקלטה עם איור קופסאות איתור של אובייקט, במקרה שלנו של שריפה\אש, או שידור של הווידאו יחד עם גילויים על הצג (שזה הפלט של קלט בשידור חי גם כן) עם אפשרות לשליחת מייל עם צירוף תמונה של האיתור לכל כתובת מייל שמגדירים.

במקרה של תמונות הפלט יהיה איתור השריפה (אם יש) וסימונה.

תיאור

שלב א':

הרצה ראשונית של התוכנה מכינה את סביבת העבודה לקראת אימון המודל שבסופה נקבל מודל מאומן על בסיס הנתונים שלנו.

חלק מהכנת סביבת העבודה כוללת, הורדת מודל קיים, התקנה של ספריות נדרשות, הורדת קבצי תמיכה המקנים כלים לאימון והגדרות למודל(פונקציות עיבוד לפני, אחרי ובזמן אימון, משקלים וערכי הייפר שנצאים ב-pipeline.config), איסוף נתונים (תמונות), בניית עותק של התמונות שהן annotated והמרתן לקבצי "TFRECORD" שהמודל יוכל ללמד מהם.

בסיום ההכנה באפשרותנו לבחור בכל מודל הנתמך בסביבה הספציפית שלנו ולהגדיר את התצורה המתאימה לנו עבור המודל שבחרנו (משתנים של המודל, פונקציות עבור הכנת נתונים למודל, אופן עיבוד של המודל, פלט והגדרת הגדרות כמו פונקציית מחיר-הפסק והגדרת מספר אובייקטים מה הם האובייקטים הללו וכו').

ההכנה הזו מאפשרת לנו להיות גמישים מבחינת רעיונות ושינויים שאנו רוצים לעשות בהמשך בלי לשנות את קוד התוכנה כמעט, אם זה שינוי מודלים, מסד הנתונים, והגדרות.

מכיוון שאימון המודל יכול לקחת המון זמן יש לנו את סביבת העבודה מוכנה להורדה (עם המודל שאנחנו אימנו עם הסביבה שאנחנו הכנו) כך שניתן לדלג על שלב ההכנה, המצריך דברים מעבר להרצת התוכנה מלכתחילה (הכנת מסד הנתונים לדוגמא).

נציין שהמודל שעבדנו איתו היה pre-trained אך לא עבור שריפות\אש.

שלב ב'

לאחר ההכנה, האימון והאימות אנו מייצאים את המודל שקיבלנו לצורך בדיקה "ידנית", משמע נריץ את המודל ונבדוק את יכולות שלו, ואכן כפי שאנחנו רואים המודל מצליח לחזות ביעילות תמונות של שריפות, יש חיזויים שבהם הוא טועה (השתקפות מסך מחשב על חלון למשל, פנסי רחוב וזריחת שמש.)

בדיקות נוספות הן, הרצת מודל עם קלט ווידאו זמן אמת (מצלמת מחשב), נדמה שאיכות ירודה של מצלמה (כמו מצלמת מחשב) פוגעת יעילות הגילוי ובאחוזי מבטח.

הרצה עם קלט ווידאו שמור נדמה שלקלט שהוא לא בזמן אמת המודל יעיל יותר (ההשערה היא שעל המחשב הפרטי כוח העיבוד והחלוקה לא מספיק טובה להבטיח שהמודל יפעל בצורה יעילה ותגלה שריפות **לפני** שיש עוד תמונה מהקלט)

המודל רץ על הFRAMES של הווידאו אחד אחד ובודק, בסוף התהליך אנו שומרים ווידאו חדש בו המודל סימן את הגילויים.

ההרצה הבאה היא דומה להרצה שלפני, רק שאנו מציגים את הגילוי בזמן אמת, משמע פתיחת חלון ווידאו לתצוגת הגילוי במקום שמירה, ההרצה הזו לא נחשבת לזמן אמת כי אנו מעבדים את הווידאו לפני ההרצה ולכן איכות הגילוי טובה, במקרה זה הוספנו את האפשרות שברגע של גילוי, אנו שולחים מייל עם התמונה בה הייתה גילוי למייל (ראה "מייל זמני" מצורף בסוף).

ההרצות הבאות הן למען בדיקה אם אפשר ליעל את מהירות העיבוד של המודל, נדמה שלשני האפשרויות (Multithreading and Multiprocessing) יש בעיות או שלא מוסיפות (בעיות טכניות הנגרמות ממערכת הפעלה, מפרט מחשב ובעיות כלליות הקיימות בספריות ובתוכנות של Python ו- python עצמה.

ראה: https://github.com/ipython/ipython/issues/12396

https://github.com/jupyter/notebook/issues/5261

(https://github.com/ipython/ipython/issues/12396

בסוף כללנו קוד להקלטה ונגינה של מצלמת רשת לצורך יצירת הקלטות ידנית.

כלים וקישורים

בהכנת סביבת העבודה אנו משתמשים במספר מקורות שבעיקרן מתבססות על מודלים הנתחמים על ידי TensorFlow:

- TensorFlow object detection API

- TensorFlow Model Garden

- Roboflow אתר לציור annotations פיצול נוח של מסד נתונים לקבוצות אימון, אימות ובחינה והמרתן ל-TFRECORD

-TensorFlow 2 Detection Model Zoo מקור המודל שלנו, שהוא **SSD MobileNet V2 FPNLite 640x640** והסיבה העיקרית לבחירת המודל היא שהיא יחסית מהירה (אך פחות מדוייקת) ויחסית "קלילה", משמע מתאימה לפיתוח עתידי להתקנת המודל על מכשירים קטנים וחסכוניים (רחפניים לדוגמא).

- early fire detection system using deep learning and opencv

- kaggle datasets בה מצאנו חלק מהתמונות כאשר גם חלקם annotated, שאר התמונות מהאתר הצריכו שימוש ב-Roboflow לצורך annotations.

הקוד וסביבת העבודה נכתבו חלקם בעזרת שני המקורות הראשונים, למשל הכנה ואתחול המודל התבססו על מקורות אלה, שאר הדברים, כמו איתור בזמן אמת ואיתור מהקלטות ושליחת הודעת התראה במייל התבססו על דברים שלמדנו במהלך הקורס ופותחו לשם המחשת היכולות של המודל, משמע יעילות איתור ,ויעילות ויכולת איתור בזמן אמת ושליח התראה.

שני הספריות הראשונות מקנות מקור לקבצים שימושיים כמו model_main_tf2.py ו- exporter_main_v2.py

ברובם הן עוזרות לטעון את ההגדרות שאנו מקנים לצורכי אימון ויצירת קובץ למודלים לצורך טעינה מהירה.

כמו כן, הספריות הללו נותנות את הפונקציות הבאות לטעינת הגדרות ואיתחול המודל שלנו.

label_map_util – טעינה תקינה של קובץ label_map המכין את האובייקטים (שמות ומס' זיהוי) שלנו

config_util – טעינה של קובץ pipeline.config המכילה את כל ההגדרות של המודל, למשל batch_size עבור האימון והאימות, פונקציות pre\postprocess וערכי hyper שאנו מגדירים עבור המודל הספציפי שלנו.

visualization_utils - פונקצייה להמרת תוצאות איתור\גילוי וציורן על תמונה נתונה.

model_builder – איתול המודל לפי המודל וההגדרות שנתנו.

כמו שהזכרנו שרוב הדברים האלה ברובם מאפשרים לשינוי סוג המודל וההגדרות בלי לשנות את הקוד הקיים כי הספריות האלה נתמכות ודואגות שאם נשתמש במודלים של TensorFlow השינויים לא יפגעו בקוד שלנו.

ספריות סטנדרטיות יותר הן:

Os - ספרייה שנותנת ממשק לפעולות של מערכת הפעלה לדוגמא פתיחת קבצים

Sys – פעולות דומות ל-OS

Numpy - ספרייה התומכת במערכים ו-ווקטורים גדולים במיוחד מבחינת מהירות חישוב, גישה ועיבוד.

Mmcv,cv2 – קריאה והצגה של קלטים אם זה תמונות או ווידאו ופעולות עלהן.

Threading – ספרייה המאפשרת הרצת חוטים

PIL – ברובו משמש לנו לעיבוד והמרת מערכים ותמונות

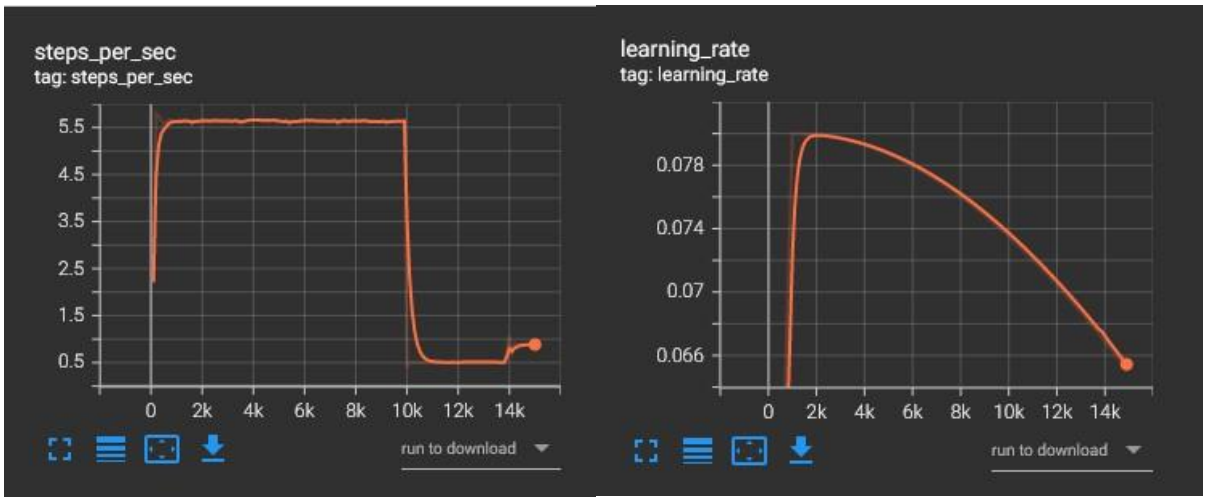
IPython - ספרייה שבחלקה משמשת עבור תמיכת מדיה וכלים של GUI

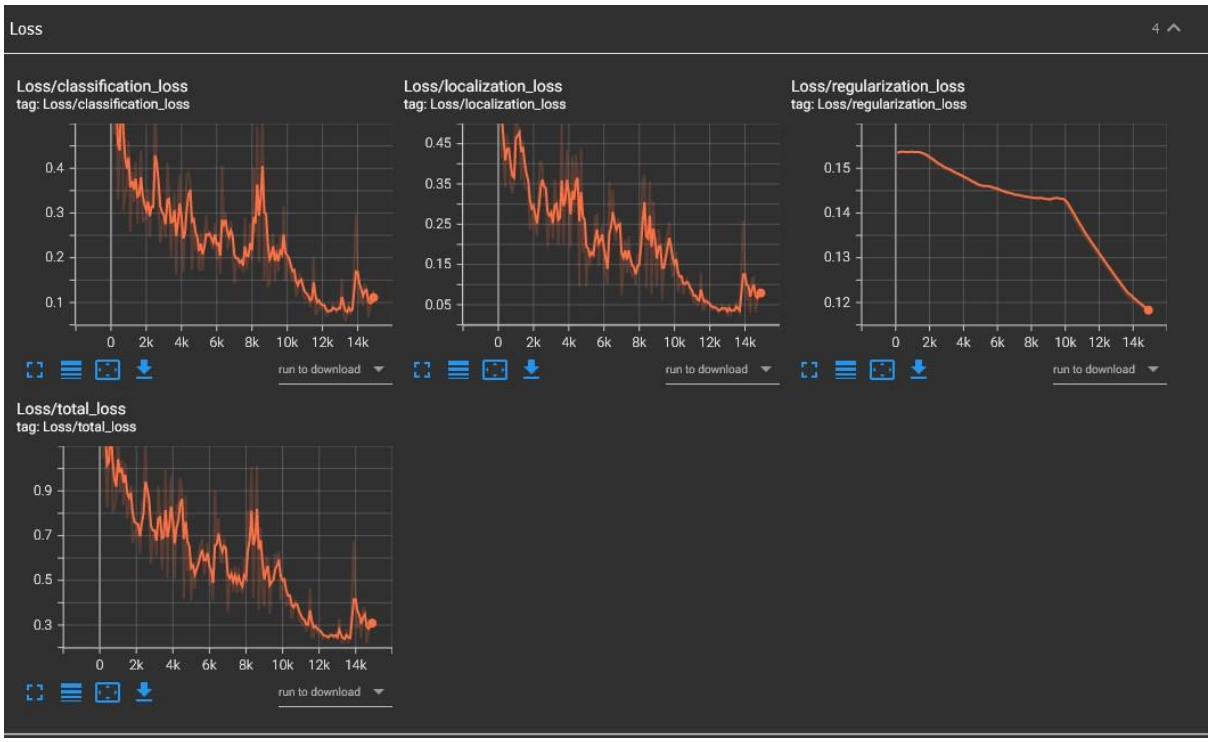
Matplotlib – ממשק לפונקציות שגורמות לעבוד כמו , matlab בעיקרון משתמשים להציג תמונות וגרפים.

Tqdm - ספרייה לתצוגת טעינה בתוך notebook

Tensorflow – משמש אותנו להגדרות סביבת המודל שלנו לשם תמיכה כללית עבורו, למשל עיקר השימוש הוא המרה של תמונות המתקבלות למשתנה הנתמך בפונקציות גילוי ועיבוד שהמודל יודע לעשות על משתנים ש- Tensorflow מעבד בשבילו.

קבצי פלט LOG של הרצת אימות בעזרת ספריית Tensorboard (validation):



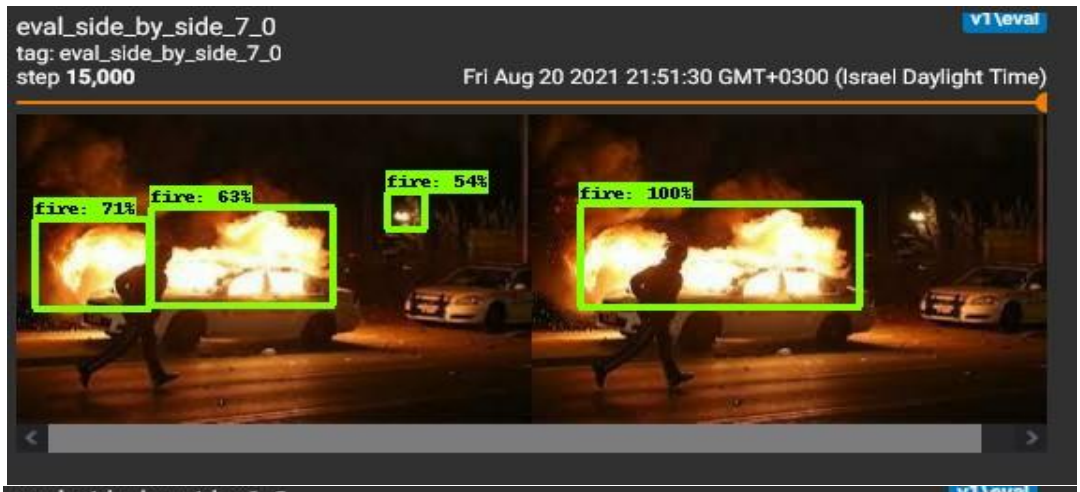


ההסבר העיקרי להתנהגות הזו היא שימוש ב-cosine_decay_learning_rate כפונקצית אימון עבור המודל שהגדרנו ב-pipeline.config שמסביר את הנפילה בחלק הסופי של האימון (האימון קרס\התנתק מספר פעמים, דבר שמסביר את הקפיצה אחרי ה-14 אלף).

ראה https://arxiv.org/abs/1908.01878 להסבר פשוט עבור cosine_decay_learning_rate.

להלן מקבץ תמונות משלב האימות (מימן) עליהם נבדק המודל (משמאל):


eval_side_by_side_8_0
tag: eval_side_by_side_8_0
step 15,000
v1\eval
Fri Aug 20 2021 21:51:30 GMT+0300 (Israel Daylight Time)
fire: 82%
EyeEm
fire: 100%
EyeEm


eval_side_by_side_7_0
tag: eval_side_by_side_7_0
step 15,000
v1\eval
Fri Aug 20 2021 21:51:30 GMT+0300 (Israel Daylight Time)
fire: 71%
fire: 63%
fire: 54%
fire: 100%


eval_side_by_side_6_0
tag: eval_side_by_side_6_0
step 15,000
v1\eval
Fri Aug 20 2021 21:51:30 GMT+0300 (Israel Daylight Time)
fire: 71%
fire: 56%
fire: 70%
fire: 100%


eval_side_by_side_4_0
tag: eval_side_by_side_4_0
step 15,000
v1\eval
Fri Aug 20 2021 21:51:30 GMT+0300 (Israel Daylight Time)
DAD CHRISTMAS
Ceremonial Lighting
DAD CHRISTMAS
fire: 100%
Ceremonial Lighting


eval_side_by_side_1_0
tag: eval_side_by_side_1_0
step 15,000
Fri Aug 20 2021 21:51:29 GMT+0300 (Israel Daylight Time)
fire: 70%
fire: 100%


eval_side_by_side_0_0
tag: eval_side_by_side_0_0
step 15,000
v1\eval
Fri Aug 20 2021 21:51:29 GMT+0300 (Israel Daylight Time)
fire: 76%
fire: 100%
fire: 100%

ראה קובץ /workspace/models/ssd_mobilenet_v2_fpnlite/v1/pipeline.config עבור הפרמטרים ופונקציות עיבוד של המודל, מקטע מהקובץ:

```
train_config {
  batch_size: 12
  data_augmentation_options {
    random_horizontal_flip {
    }
  }
  data_augmentation_options {
    random_crop_image {
      min_object_covered: 0.0
      min_aspect_ratio: 0.75
      max_aspect_ratio: 3.0
      min_area: 0.75
      max_area: 1.0
      overlap_thresh: 0.0
    }
  }
  sync_replicas: true
  optimizer {
    momentum_optimizer {
      learning_rate {
        cosine_decay_learning_rate {
          learning_rate_base: 0.07999999821186066
          total_steps: 50000
          warmup_learning_rate: 0.026666000485420227
          warmup_steps: 1000
        }
      }
      momentum_optimizer_value: 0.8999999761581421
    }
    use_moving_average: false
  }
  fine_tune_checkpoint: "/content/workspace/pre_trained_models/ssd_mobilenet_v2_fpnlite_640x640_coco17_tpu-8/checkpoint/ckpt-0"
  num_steps: 50000
  startup_delay_steps: 0.0
  replicas_to_aggregate: 8
  max_number_of_boxes: 10
  unpad_groundtruth_tensors: false
  fine_tune_checkpoint_type: "detection"
  fine_tune_checkpoint_version: V2
}
train_input_reader {
  label_map_path: "/content/workspace/data/label_map.txt"
  tf_record_input_reader {
    input_path: "/content/workspace/data/train.record/trainAnno.tfrecord"
  }
}
eval_config {
  metrics_set: "coco_detection_metrics"
  use_moving_averages: false
}
eval_input_reader {
  label_map_path: "/content/workspace/data/label_map.txt"
  shuffle: false
  num_epochs: 1
  tf_record_input_reader {
    input_path: "/content/workspace/data/validation.record/trainAnno.tfrecord"
  }
```

קובץ labal_map.txt די פשוט כי הרי אנו מחפשים דבר אחד לאיתור

```
item {
        id: 1
        name: "fire"
}
item {
        id: 2
        name: "natural"
}
```

רעיונות לשיפורים אפשריים: ביצוע העבודה הראתה לנו את הקשיים והבעיות הקיימות באימון מודל כזה, המודל שאמור להיות מהיר מקרטע אך עושה את העבודה בזמן אמת וחוסר תמיכה וחוסר משאבים מותאמים גורם לכך שאי אפשר לשפרם, כך שהשיפור האפשרי הראשון הוא בנייה מורכבת יותר של הפרויקט התומכת ב- multiprocessing שיאיץ את הביצועים, שימוש במודל פשוט יותר (אם זה מודל שלא מאומן מהתחלה או מודל "קליל" יותר מהמודל הנוכחי).

באפשרות התוכנה לשלוח הודעת מייל במקרה של גילוי שריפה, ניתן להוסיף GPS ופעולות רחפן הסורק שטחים (ניתן להוסיף שניתן לאמן את המודל לחזות לפי המיקום, מה הסיכויים לשריפה בזמן ובמצב האקלים הנתון ושיבחר אם להמשיך לשהות במקום זה או למקום אחר עם סיכויים גבוהים יותר לשריפות) שמרחף אל אזורי שריפה שגילה, לצלם ולשלוח מייל עם כאורדינטות ה-GPS של המיקום שלו, דבר הדורש חומרה (GPS ורחפן).

אימון יעיל יותר משמע יותר תמונות, במיוחד עם שילוב עם תמונות תרמיות במקומות, דרך אחד לייעול הוא שימוש במלוא המודל(pre-trained) ולפסול אובייקטים, למשל ניתן לפסול אותם כאשר ההערכה של המודל כוללת אש+פנסים(בהנחה שהמודל הpre-trained יודע פנסים באופן יעיל כבר).

קישור להקלטה הרצת קוד + הצגת מצגת:

הרצת קוד : https://youtu.be/qpIQP3RWE4w

הצגת מצגת: https://youtu.be/Ev2kBvPrwn0

קישור לקוד וסביבת עבודה:

https://github.com/Mosa-T/Project_deep_learning

ביבליוגרפיה:

ראה קישורים בכלים וקישורים

"מייל זמני":