מעבדה 1-2

טכניקות אימון בסיסיות (מתדולוגיה) ומתקדמות, ואימון רשת לזיהוי אוביקטים.

1. דטא: בעולם רשתות הניורונים הדבר אולי הכי חשוב שבלעדיו שכנראה לא היינו פה היו הוא הדטא.  
   כאשר נבחר/נאסוף דטא נרצה לעשות זאת בקפידה ועל פי סטנדרטים שנקבע. ככל שהדטא שלנו יהיה "טוב" יותר כך גובר הסיכוי שהרשת שלנו תלמד טוב יותר (כמובן שזה לא מובטח).  
   כאשר מדברים על דטא אז עולות שאלות כמו: כמה דטא נצטרך? איזה איכות של תמונות/סרטונים נרצה וכ'ו.
   1. כמות הדטא: כמות הדטא יכולה חד משמעית להשפיע על תוצאות המודל. מודל שלא אומן על מספיק דטא כנראה לא יצליח להכליל את הבעיה יותר מידי ואז הוא ככל הנראה יכנס לoverfit (לא בהכרח אבל סיכוי לא קטן). לעומת זאת, אם יש הרבה דטא אז המודל יראה יותר דוגמאות ולכן התוצאה ככל הנראה תהיה יותר כללית.  
      מה המשמעות מבחינת בחירת ארכיטקטורה כאשר יש או אין מספיק דטא?
   2. איכות הדטא: כאשר אנו מדברים על איכות דטא אז אפשר לחשוב על רזולוציה של התמונות, נקיון התמונות וכ'ו.  
      ככל שהתמונות ברזולוציה גבוהה יותר כך יש בהן יותר מידי, אבל הרזולוציה כשלעצמה איננה אינדיקציה לבד לאיכות התמונה.  
      את איכות התמונה יכול לקבוע אדם.  
      נקיון התמונה מתייחס לכמה התמונה אינפורמטיבית והאם יש בא הרבה רעש או לא – את זה נרחיב בכיתה.
   3. חלוקת הדטא: גם זהו חלק קריטי שצריך להחליט. אם יש מספיק תמונות אז כמה נחלק לvalidation וכמה לtest או שפשוט נעשה רק train + val. אחוז החלוקה לרוב משתנה בהתאם לכמות הדטא!
2. ארכיטקטורה: לכל רשת ישנה ארכיטקטורה משלה, אם זה כמה שכבות קנובולוציה ואז שכבה לינארית או כל מיני שילובים אחרים, למשל, ResNet50, VGG19 וכו'.  
   לכל ארכיטקטורה יש את היתרונות וחסרונות שלה.
   * 1. בחירת ארכיטקטורה:
        1. בד"כ נבחר על פי ההצלחה של הרשת במדדים, למשל עבור סיווג נרצה לבחור רשת שקיבלה תוצאה יחסית גבוה בעבר על דטה סט כלשהו(למשל imagenet).
        2. בהתאם לכמות הדטא שלנו נבחר את הארכיטקטורה שלפי דעתנו כמות הפרמטרים לא תהיה יותר מידי ולא קצת מידי. למשל עבור דטא-סט קטן לא נבחר בResNet151 או EfficenNetB7 כמו שבדטא סט גדול לא נבחר ResNet18 או EfficentNetB0. הבחירה הזו עלולה להביא את המודל שלנו או למצב של OverFit או UnderFit.
        3. בבחירה של ארכיטקטורה צריך גם לקחת בחשבון את הנתונים הפיזים של המכונה שלנו(זכרון, מעבד, כרטיס גרפי) ואז ישנו משחק בין גודל הארכיטקטורה לבין גודל התמונה וגודל הבאץ' של התמונות שנדבר על זה מטה.
3. היפר-פרמטרים: מהקורס הקודם הכרנו כל מיני פרמטרים שמשפיעים על האימון כמו:
   1. קצב הלמידה – LR
   2. גודל הבאץ' – Batch size
   3. למשל Dropout probability , Image resolution etc.

כל היפר פרמטר יכול להשפיע על הלמידה בצורה דרמטית או כמעט לא להשפיע בכלל.

כאשר נרצה לנסות היפר פרמטר חדש יהיה עדיף לנסות לשנות רק היפר פרמטר אחד ולא כמה בבת אחת. וגם כאשר נשנה כדאי לתעד בדיוק מה השתנה כדי שנוכל לשחזר את התוצאות!.

1. פונקצית הפסדים:  
     
   פונקציית לוס היא אחד הדברים הקריטים כאשר מאמנים, הבחירה של הפונקציית צריכה להיות בהתאם לבעיה שאנחנו רוצים לפתור למשל : Classification : Cross entropy, Segmentation: Dice, Regression: Mean Squared Error.  
   ניתן גם לשלב כמה סוגים של לוסים, זהו דבר מקובל שעושים לא מעט.
2. מדדים:  
     
   אז איך אנחנו יודעים בסוף אם המודל שלנו הצליח או לא? פשוט נוסיף מדדים שיעזרו לנו להבין את זה. וכמו בסעיף 4, גם כאן עבור כל בעיה יש את המדד שלה כמו: Classification: Accuracy, Precision, etc..  
   Segmentation: IOU, Object-Detection: Average Precision, Average Recall, IOU etc..  
   וכמובן גם נרצה להסתכל על הלוסים שנקבל שזה בעצם גם מדד סופר חשוב.

אז לאחר שבחרנו והכנו את כל הדברים שבסעיפים למעלה, הגיע הזמן לאמן.  
גם פה ישנה צורה ודרך טובה ומקובלת איך לעשות כמו שצריך.

1. Dirty implementation:  
   לכאורה הדרך הכי טובה היא לעשות מימוש מהיר ו"מלוכלך" כאשר פה הכוונה היאפשוט לקחת את המודל כמו שהוא עם ההגדרות הדיפולטיביות ומינימום שינוים בלי אוגמנטציות על הדטא למעט הדברים ההכרחיים כמו שינוי גודל התמונה במקרה שלא מצליחים לאמן.  
   פשוט מריצים את זה ככה ורואים את התוצאות הראשוניות, מנתחים את התוצאות, הגרפים והמדדים.  
   ננסה להבין איפה המודל טעה, באיזה תמונות דווקא שם הוא טעה.
2. שינויים קטנים:  
   בהתאם לתוצאות וניתוח התוצאות שלנו ננסה להתאים ולשנות קצת פרמטרים, כל פעם קצת כדי שנוכל להצליח לעקוב.
3. ניתוח התוצאות: נסתכל שוב על המדדים והתוצאות שקיבלנו מהאימון השני לאחר הdirty implementation ונחקור את מה שקיבלנו, נסתכל על המדדים השונים ממש כמו בהתחלה.
4. וככה נעשה את 2-3 בגדול עד שנקבל תוצאות שיתאימו לנו.

בעיות שעלולות לצוץ בזמן אימון:

1. Overfit – כאשר המודל שלנו "מוצלח" מאוד על הדטא של האימון ופחות מוצלח על הטסט או ולידציה.  
   בד"כ אפשר לראות זאת לפי המדדים באימון שהם ככל הנראה ימשיכו להשתפר לעומת המדדים על הVAL שהם לא ישתפרו או יעשו גרועים יותר.  
   נדבר על שיטות להילחם בזה בשיעור.
2. Underfit -כאשר המודל שלנו לא מצליח ללמוד בכלל והמדדים פשוט נשארים נמוכים.
3. Exploding Gradients - כאשר הגרדיאנטים שלנו גדולים מידי..
4. Vanishing Gradients – כאשר הגרדיאנטים שלנו קטנים מאוד עד כדי שהם שואפים ל0 וכמעט לא משפיעים על האימון.

אימון רשת לזיהוי אוביקטים – מעבדה:  
הקדמה בע"פ

1. הכנת הדטא:
   1. ברשותכם 200 תמונות של אנשים עם מסיכות ובלי מסיכות. עליכם לתייג את התמונות ולשמור את המטהדטא לפי פורמט CoCo. כאן אנו נשתמש בכלי קל לביצוע תיוגים:  
      <https://github.com/jsbroks/coco-annotator> כלי תיוג  
      <https://drive.google.com/drive/folders/1Fxe7ON3877UB3ChIKHfoKCzKllJtIOfW?usp=sharing> דטה-סט.
   2. הציגו את התמונות השונות (עם מסיכה, בלי מסיכה) ותעשו ניתוח סטטיסטי לדטא(כמות התמונות עם מסיכה לעומת בלי) גודל תמונות ממוצע וכו'.
      1. פרטו האם לדעתכם עקב ההתפלגות התמונות עלולות להיווצר בעיות בזמן האימון?
   3. בחרו את הארכיטקטורה שלכם לביצוע הזיהוי והסבירו את הבחירה תוך כדי פירוט שכולל שיקולים מקצועיים.
      1. ביחרו את הBackbone עבור ארכיטקטורת הזיהוי והסבירו את בחירתכם.
   4. הכינו “dirty implementation” של המודל שבחרתם, בצעו אימון בלי שינוים גדולים של הפרמטרים במהלך 10 אפוקים והציגו את התוצאות (מדדים רלבנטיים, גרפים של האימון וולידציה ותמונות בהן המודל הצליח ותמונות בהן לא הצליח ונסו להסיק מסקנות מדוע לא עבד ואולי על מה המודל "התבלבל").
   5. ביחרו 2-3 היפר פרמטרים שונים, שנו אותם ואמנו 10 אפוקים.   
      נתחו את התוצאות כמו בסעיף הקודם. הסבירו את בחירתכם לגבי הפרמטרים.
   6. ביחרו את הBackBone הכי גדול שניתן ונסו לאמן כמו בסעיפים לעיל, נתחו את התוצאות והסבירו האם המודל נכנס לOverFit או UnderFit.