9. Computer Vision

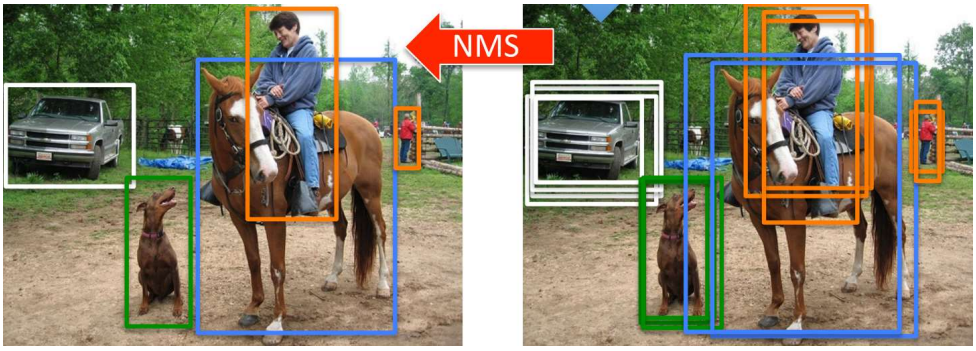
9.1 Object Detection

9.1.2 You Only Look Once (YOLO)

עד שנת 2016 כל הגלאים (detectors) המבוססים על למידה עמוקה (כמו למשל R-CNN family) היו גלאים דו-שלביים – השלב הראשון יצר אלפי הצעות (proposals) למסגרות מלבניות החשודות כמכילות אובייקטים, ואלו הוזנו בזו אחר זו לשלב השני אשר דייק את המסגרות וביצע סיווג לאובייקטים המוכלים בהן.
ארכיטקטורת YOLO, הנגזרת מראשי התיבות של YOU ONLY LOOK ONCE, הוצגה על ידי ג'וזף רדמון ב-2016, והייתה הגלאי הראשון שמורכב משלב יחיד, ובו הרשת מנבאת את המסגרות וגם מסווגת את האובייקטים שבתוכן בבת אחת. בנוסף, ארכיטקטורת YOLO מתאפיינת במיעוט פרמטרים וכמות פעולות אריתמטיות. היא אמנם משלמת על המבנה הרזה והאלגנטי שלה בדיוק נמוך יותר, אך המהירות הגבוהה (שנובעת בעיקר מהיעדר האילוץ לעבד מסגרת יחידה בשלב השני בכל מעבר של הלולאה) הפכה את גישת השלב היחיד לאטרקטיבית מאוד, במיוחד למעבדים קטנים כדוגמת מכשירי mobile. בעקבות עבודה זו פותחו גלאים רבים על בסיס שלב יחיד, כולל גרסאות מתקדמות יותר של YOLO (הגרסה המתקדמת ביותר כיום היא v5).

NMS (Non-Maximum Suppression)

כמעט כל אלגוריתם של זיהוי אובייקטים מייצר מספר רב של מסגרות חשודות, כאשר רובן מיותרות ויש צורך לדלל את מספרן. הסיבה ליצירת מספר גדול של מסגרות נובע מאופי פעולת הגלאים. בזכות התכונות הלוקאליות של פעולת הקונבולוציה, מפת הפיצ'רים במוצא הגלאי ניתנת לתיאור כמטריצת משבצות כאשר כל משבצת שקולה לריבוע של הרבה פיקסלים בתמונה המקורית. רוב הגלאים פועלים בשיטת עוגנים (anchors), כאשר כל משבצת במוצא הגלאי מנבאת מספר קבוע של מסגרות שעשויות להכיל אובייקט (למשל, ב-YOLOv2 המספר הוא 5, ובגרסאות המתקדמות יותר המספר הוא 3). השיטה הזו יוצרת אלפי מסגרות שרק מעטות מהן הן משמעותיות. בנוסף – יש ריבוי של מסגרות דומות בסביבת כל אובייקט (למשל – YOLOv3 מנבאת יותר מ-7000 מסגרות לכל תמונה). אחת הדרכים הפופולריות לסנן את אלפי המסגרות ולהשאיר רק את המשמעותיות נקראת NMS. בשיטה זו מתבצעת השוואה בין זוגות של קופסאות מאותה המחלקה (למשל – חתול), ובמידה שיש ביניהן חפיפה גבוהה – מוחקים את המסגרת בעלת הוודאות הנמוכה היותר ונשארים רק עם המסגרת בעלת רמת הוודאות הגבוהה. שיטה זו בזבזנית בחישוב (סיבוכיות פרופורציונלית לריבוע מספר המסגרות) ואינה חלק מהמודל המתאמן, אך עם זאת הינה אינטואיטיבית יחסית למימוש, ומשום כך נמצאת בשימוש נפוץ בגלאים, כולל ב-YOLO.

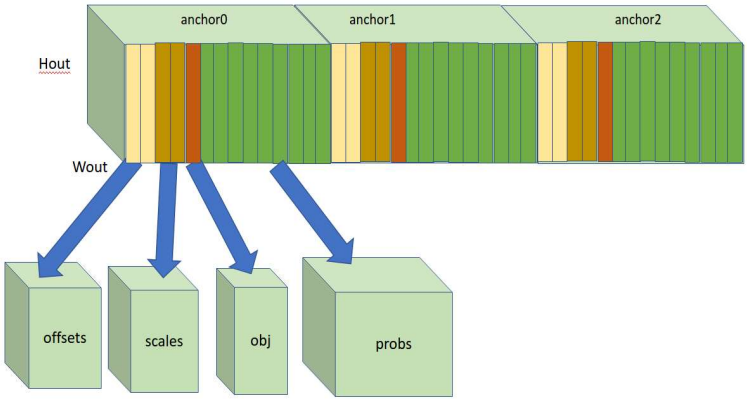


איור 9.1 אלגוריתם NMS.

YOLO Head

כמו רוב הגלאים, YOLO הינו מבוסס-עוגנים (anchor-based), שהינן תיבות מלבניות קבועות ושונות זו מזו בצורתן. לכל עוגן מוקצה מקטע של פיצ'רים במפת המוצא של הרשת וכל הניבויים במקטע הזה מקודדים כסטיות (offsets) ביחס לממדי העוגן. כפי שניתן לראות באיור 9.2, הפיצ'רים של כל תא מרחבי במפת המוצא מחולקים למקטעים על פי העוגנים (שלושה עוגנים במקרה הזה).
ניבוי הוא מסגרת שמרכזה נמצא בנקודה כלשהי בשטח התא (השקול לריבוע של מספר פיקסלים בתמונה המקורית). ההסטה המדויקת של מרכז המסגרת ביחס לתא ניתנת על ידי שני הפיצ'רים הראשונים ברצף. לוג המימדים של הקופסא (ביחס לממדי העוגן) ניתן באמצעות שני הפיצ'רים הבאים ברצף. הפיצ'ר החמישי לומד את מידת ה-objectness, כפי שהוסברה לעיל. שאר הפיצ'רים ברצף של העוגן הנ"ל הם ההסתברויות המותנות לכל מחלקה (אם

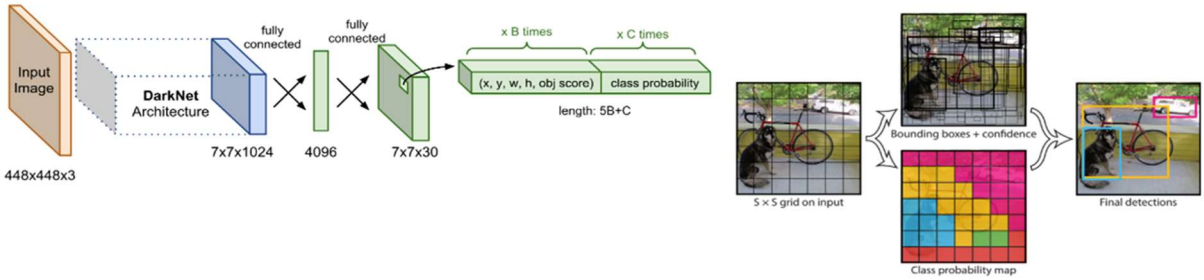
אוסף הנתונים מכיל 80 מחלקות, יהיו 80 פיצ'רים כאלה). על מנת לקבל רמת ודאות סופית, יש להכפיל את מדד ה-objectness במדד ההסתברות המותנה לכל מחלקה.



איור 9.2 ראש YOLO.

YOLOv1

מודלי YOLO מבוססים על גרסאות Backbone הנקראות Darknet ומשמשות לעיבוד פיצ'רים מתוך התמונה, ומראש detection המקבל את הפיצ'רים האלה ומתאמן לייצר מהם ניבויים למסגרות סביב אובייקטים.
המודל מחלק את התמונה לרשת בעלת $S \times S$ משבצות, כאשר כל משבצת מנבאת N מסגרות של אובייקטים בשיטת העוגנים, כאמור לעיל. כל ניבוי כולל את מספר ערכים: הסטת הקואורדינטות x, y של מרכז המסגרת ביחס למשבצת, הגובה והרוחב של המסגרת, ורמת ה-objectness, כפי שהוסברה לעיל. בנוסף, כל מסגרת מבצעת גם סיווג, כלומר מנבאת את רמת הוודאות של השתייכות האובייקט לכל אחת מהמחלקות האפשרויות. החידוש באלגוריתם נעוץ בעובדה שחיזוי המסגרות וסיווגן לאובייקטים נעשה במקביל, ולא באופן דו-שלבי. הרעיון הוא להתייחס לסוג האובייקט כעוד פיצ'ר שהרשת מנסה לחזות בנוסף למיקום וגודל של המסגרת.

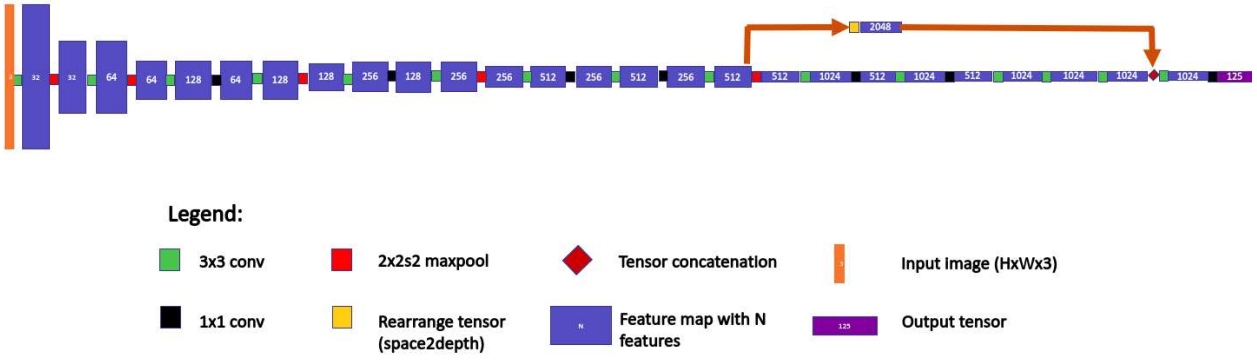


איור 9.3 ארכיטקטורת YOLO.

הגרסה הראשונה של ארכיטקטורת YOLO כללה שכבת fully connected שהוסרה בגרסאות הבאות. בנוסף, פונקציית ה-LOSS וסדר התכונות של המסגרות במוצא הרשת השתנו, אבל הרעיון נותר זהה.

YOLOv2

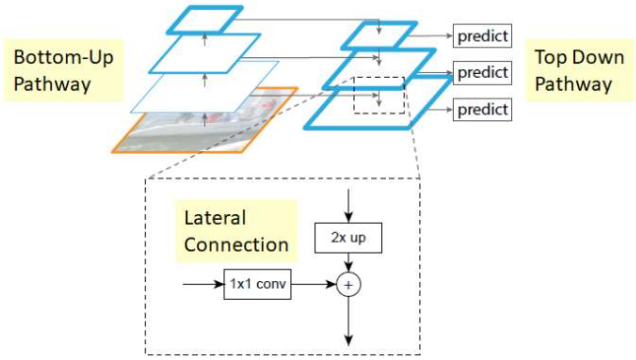
גרסה זו משתמשת ברשת Backbone הנקראת Darknet19 ובה 19 קונבולוציות. המודל כולו כולל 22 קונבולוציות (מלבד 5 שכבות ה-MAXPOOL המקובלות על מנת למצוא את הפיצ'רים), ועוד מסלול עוקף בסמוך לסוף הרשת המחזק את יכולת העיבוד. הכותב של המאמר המקורי, רדמון, נהג לפתח גם גרסאות "Tiny" לכל מודל. הווריאנט YOLOv2 Tiny מכיל רשת Backbone קצרה יותר וללא מסלול עוקף ומבנהו לינארי ופשוט מאוד. ביצועיו אמנם נמוכים יותר אך הוא מהיר מאוד (207 תמונות לשנייה לעומת 67 של מודל YOLOv2, על מעבד TitanX).
מעניין לציין ש-YOLOv2 הוא המודל הראשון שאומן על תמונות בממדים משתנים, תהליך המשפר את דיוק המודל.



איור 9.4 ארכיטקטורת YOLOv2.

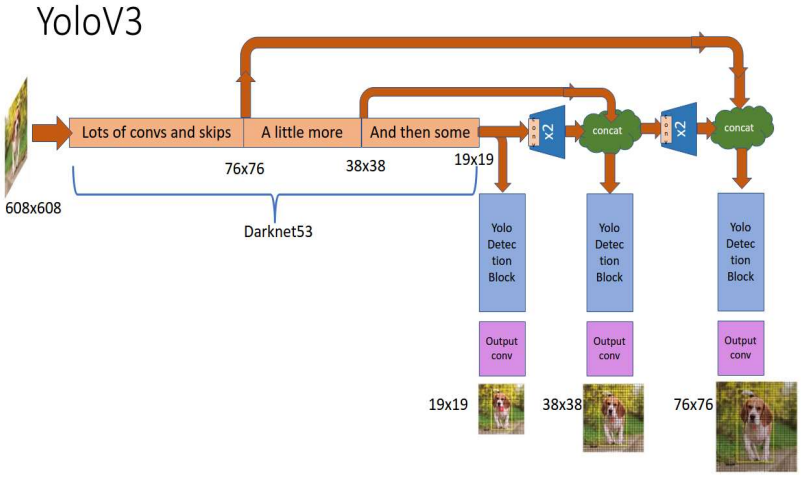
YOLOv3

כל דור נוסף של YOLO הציג חידושים ארכיטקטוניים שהגדילו את מורכבות החישוב וגודל המודל ושיפרו את ביצועיו. גרסה מספר 3 מבוססת על רשת Backbone גדולה בהרבה שנקראת Darknet53 ובה, כפי שעולה משמה, 53 קונבולוציות. כמו כן הרשת מכילה צוואר של ארכיטקטורת Feature Pyramid Network (FPN) בעלת שלושה ראשי גילוי, כאשר כל אחד מהם הוא בעל רזולוציה שונה ($19 \times 19, 38 \times 38$ ו-76×76).
ארכיטקטורת FPN היא תוספת בקצה ה-Backbone, אשר מגדילה חזרה באופן הדרגתי את מפת הפיצ'רים. תוספת זו נועדה ליצור מסלול Top Down במקביל למסלול ה-Feed Forward ברשת ה-Backbone, שבנוי למעשה בצורת Bottom Up, בו ככל שמתקדמים בשכבות ה-Backbone מתבצע עיבוד ברמה גבוהה יותר והרזולוציה המרחבית של מפת הפיצ'רים הולכת וקטנה. בעזרת השימוש ברשת ה-FPN המודל לומד לנצל את המיטב משני עולמות: הוא משתמש במידע שטמון במפת הפיצ'רים הגדולה, שהיא אמנם פחות מעובדת אך בעלת פיצ'רים ברזולוציה מרחבית גבוהה, ובנוסף הוא מנצל גם את המידע ממפת הפיצ'רים הקטנה, שהיא אמנם בעלת פיצ'רים ברזולוציה מרחבית נמוכה, אך עם זאת היא מעובדת יותר.
לאחר כל הגדלה של מפת הפיצ'רים מתבצע חיבור בין התוצאה לבין מפת פיצ'רים קדומה יותר בממדים זהים (מתוך ה-Backbone). זאת בדומה לחיבורים העקיפים ברשת ResNet המסייעים להתכנסות האימון. השכבות השונות של רשת FPN מאפשרות לגלאי למצוא מיקום מדויק יותר של האובייקט ברזולוציות השונות, מה שמעניק לרשת זו את היכולת להבחין באובייקטים קטנים בתמונה גדולה.



איור 9.5 ארכיטקטורת Feature Pyramid Network (FPN), המשלבת מסלול top down לאחר ה-bottom up.

לראש המודל של YOLOv3 יש מספר ענפי detection, אשר כל אחד מהם פועל על מפת פיצ'רים ברזולוציה שונה ובאופן טבעי מתמחה בגילוי אובייקטים בגודל שונה (הענף בעל הרזולוציה הנמוכה מתמחה בגילוי אובייקטים גדולים, והענף בעל הרזולוציה הגבוהה מתמחה בגילוי אובייקטים קטנים).



איור 9.6 ארכיטקטורת YOLOv3.

YOLOv4

רשת YOLOv4 היא בעלת ראש זהה לזה של שתי הגרסאות הקודמות, אך ה-Backbone שונה ומורכב יותר. הוא נקרא CSPDarknet53 כאשר CSPNET היא קיצור של Cross-Stage Partial Network. רשת זו מפצלת מפות פיצ'רים לטובת קונבולוציה בחלקים ואיחוד מחדש. פיצול זה מאפשר, כמוזכר במאמר המקורי, חלחול טוב יותר של הגרדיאנטים בשלב האימון. בנוסף, נעשה ברשת זו שימוש בפונקציית אקטיבציה הנקראת Mish (ולא Leaky ReLU כמו בגרסאות הקודמות).
אנקדוטה מעניינת – החל מגרסה זו התצורות אינן של היוצר המקורי ג'וזף רדמון, שהחליט לפרוש ממחקר ראייה ממוחשבת בגלל שיקולים אתיים של שימושים צבאיים או שימושים הפוגעים בפרטיות.

YOLOv5

רשת YOLOv5 מוסיפה עוד שכלולים על רשת ה-Backbone על פני זו של הדור הקודם, ומציגה אופרטור חדש המארגן פיקסלים סמוכים בתמונת במימד הפיצ'רים. אופרטור זה דואג לכך שהכניסה לרשת היא לא בעומק 3 פיצ'רים כמקובל (RGB), אלא 12 פיצ'רים, תוך הקטנת המימד המרחבי. באופן זה הרשת מתאמנת לעבד תמונות ברזולוציה גבוהה, ואף לזהות אובייקטים גדולים בקלות רבה יותר, שכן שדה התמך (receptive field) של הקונבולוציות מכיל מידע משטח תמונה גדול יותר.

9.2 Segmentation

אחד האתגרים הכי משמעותיים בעולם הראייה הממוחשבת הוא זיהוי אובייקטים בתמונה והבנת המתרחש בה. אחת הטכניקות הקלאסיות להתמודדות עם משימה זו הינה ביצוע סגמנטציה, כלומר, התאמת label לכל פיקסל בתמונה. בתהליך הסגמנטציה מבצעים חלוקה/בידול בין עצמים שונים בתמונה המצולמת באמצעות סיווג ברמת הפיקסל, כלומר כל פיקסל בתמונה יסווג וישויך למחלקה מסוימת.

ישנם שימושים מגוונים באלגוריתמים של סגמנטציה – הפרדה של עצמים מסוימים מהרקע שמאחוריהם, מציאת קשרים בין עצמים ועוד. לדוגמה, תוכנות של שיחות ועידה, כמו zoom, skype, teams וכדו', מאפשרות בחירת רקעים שונים עבור המשתמש, כאשר מלבד הרקע הנבחר רק הגוף של המשתתף מוצג בווידאו. הפרדת גוף האדם מהרקע והטמעת רקע אחר מתבצעות באמצעות אלגוריתמים של סגמנטציה. דוגמה נוספת – ניתן לזהות בתמונה אדם, כלב, וביניהם רצועה, ומכך ניתן להסיק שתוכן התמונה הוא אדם מחזיק כלב בעזרת רצועה. במקרה זה, הסגמנטציה נועדה למצוא קשר בין עצמים ולהבין את המתרחש.

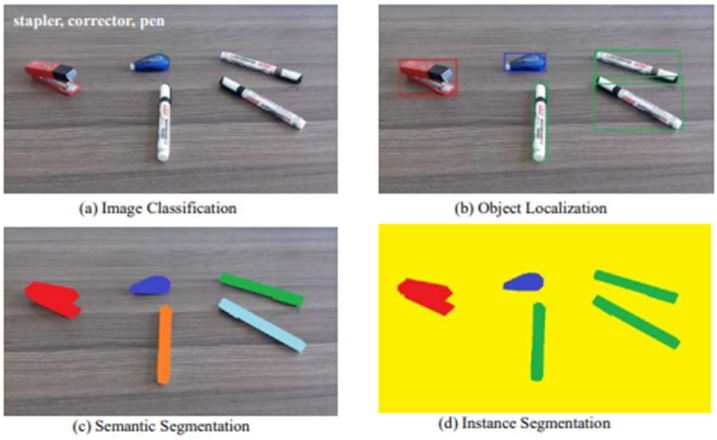
9.2.1 Semantic Segmentation vs. Instance Segmentation

קיימים שני סוגים עיקריים של סגמנטציה:

Semantic segmentation (חלוקה סמנטית) - חלוקה של כל פיקסל בתמונה למחלקה אליה העצם אותו הוא מייצג שייך. למשל, פיקסל יכול להיות משויך לכלי רכב, בן אנוש, מבנה וכדו'.

Instance segmentation (חלוקה מופעית) - חלוקה של פיקסל בתמונה למופע של אותה מחלקה אליה העצם אותו הוא מייצג שייך. במקרה זה, בתמונה בה מופיעים מספר כלי רכב, תבוצע חלוקה של כל פיקסל לאיזה כלי רכב אותו פיקסל מייצג – מכונית 1, מכונית 2, אופנוע 1, משאית 1 וכדו'.

ההבדל העיקרי בין שני סוגים אלו הוא ברמת עומק המיפוי של פיקסל - המיפוי עשוי לסווג את הפיקסל למחלקה כלשהי, או לעצם ספציפי בתמונה. עומק המיפוי משליך גם על עלות המיפוי. החלוקה הסמנטית מבוצעת ישירות, בעוד שהחלוקה המופעית דורשת בנוסף ביצוע של זיהוי אובייקטים כדי לסווג מופעים שונים של המחלקות.



איור 9.7 משימות שונות תחת התחום של Computer Vision. ניתן להבחין בהבדל שבין Semantic segmentation (התאמת כל פיקסל למחלקה מסוימת) לבין Instance segmentation (התאמת כל פיקסל למופע של מחלקה מסוימת).

9.3 Face Recognition and Pose Estimation

9.3.1 Face Recognition

אחד מהיישומים החשובים בעיבוד תמונה הינו זיהוי פנים, כאשר ניתן לחלק משימה זו לשלושה שלבים:

1. Detection – מציאת הפרצופים בתמונה.
2. Embedding – מיפוי כל פרצוף למרחב חדש, בו המאפיינים שאינם קשורים לתיאור הפנים (למשל: זווית, מיקום, תאורה וכדו') אינם משפיעים על הייצוג.
3. Searching – חיפוש במאגר של תמונות למציאת תמונת פנים הקרובה לתמונת הפנים שחולצה מהתמונה המקורית.

גישה פשטנית, כמו למשל בניית מסווג המכיל מספר יציאות כמספר הפנים אותם רוצים לזהות, הינה בעייתית משתי סיבות עיקריות: ראשית יש צורך באלפי דוגמאות לכל אדם (שלא ניתן בהכרח להשיג). כמו כן, נצטרך ללמד את המערכת מחדש בכל פעם שרוצים להוסיף מישהו חדש. כדי להתגבר על בעיות אלו מבצעים "למידת מטריקה" (metric learning) בה מזקקים מאפיינים של פנים ויוצרים וקטור יחסית קצר, למשל באורך 128, המכיל את האלמנטים המרכזיים בתמונת הפנים. כעת נפרט את שלושת השלבים:

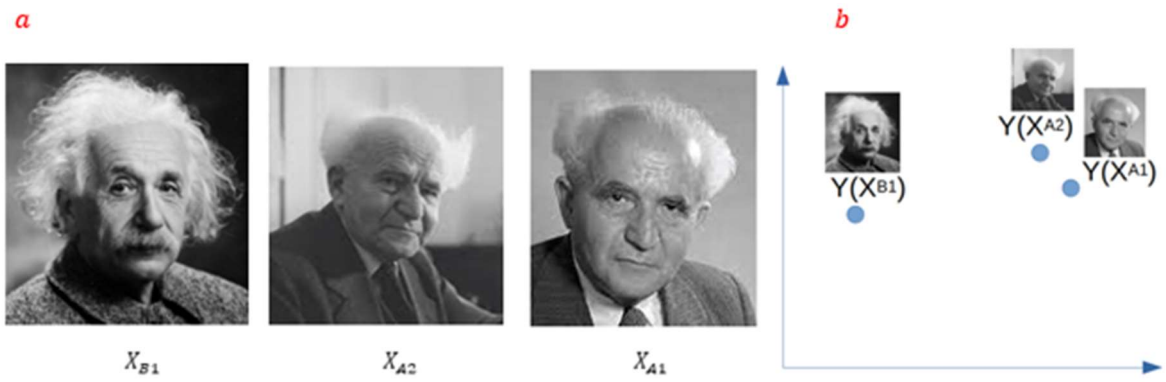
1. מציאת פנים.

כדי למצוא פרצופים בתמונה ניתן להשתמש ברשתות המבצעות detection, כפי שתואר בפרק 9.1. שיטה מקובלת למשימה זו הינה Yolo, המבוססת על חלוקת התמונה למשבצות, כאשר עבור כל משבצת בוחנים האם יש בה אובייקט מסוים, מהו אותו אובייקט, ומה ה-bounding box שלו.

2. תיאור פנים.

כאמור, המשימה בתיאור פנים נעשית בעזרת metric learning, כאשר הרעיון הוא לזקק פנים לוקטור שאינו מושפע ממאפיינים שלא שייכים באופן מהותי לפנים הספציפיות האלה, כגון זווית צילום, רמת תאורה וכדו'. בכדי לעשות זאת יש לבנות רשת המקבלת פנים של בנאדם ומחזירה וקטור, כאשר הדרישה היא שעבור שתי תמונות של אותו אדם יתקבלו וקטורים מאוד דומים, ועבור פרצופים של אנשים שונים יתקבלו וקטורים שונים. למעשה, פונקציית ה-loss תקבל בכל פעם minibatch, ותעניש בהתאם לקרבה בין וקטורים של אנשים שונים וריחוק בין וקטורים של אותו אדם.

כעת נניח שיש לנו קלט X, המכיל אוסף פרצופים. כל איש יסומן באות אחרת – A,B,C, ותמונות שונות של אותו אדם יסומנו על ידי אות ומספר, כך שלמשל X_{A1} זוהי התמונה הראשונה של אדם A בסט הקלט X, וכמובן ש-X_{A1} ו-X_{A2} הן שתי תמונות של אותו אדם. באופן גרפי, בדו-ממד ניתן לתאר זאת כך (בפועל הווקטורים המייצגים פנים יהיו בממד גבוה יותר):



איור 9.8 a) דוגמאות מסט הפרצופים X. b) איך נרצה שהדאטה ימופה לממד חדש Y.

כאמור, נרצה לבנות פונקציית loss שמעודדת קירבה בין X_{A1} ו-X_{A2}, וריחוק בין X_{A1} ו-X_{B1}. פונקציית ה-loss מורכבת משני איברים, המודדים מרחק אוקלידי בין וקטורים שונים:

$$L = \sum_{X} \left\| Y(X^{Ai}) - Y(X^{Aj}) \right\| - \left\| Y(X^{Ai}) - Y(X^{Bj}) \right\|$$

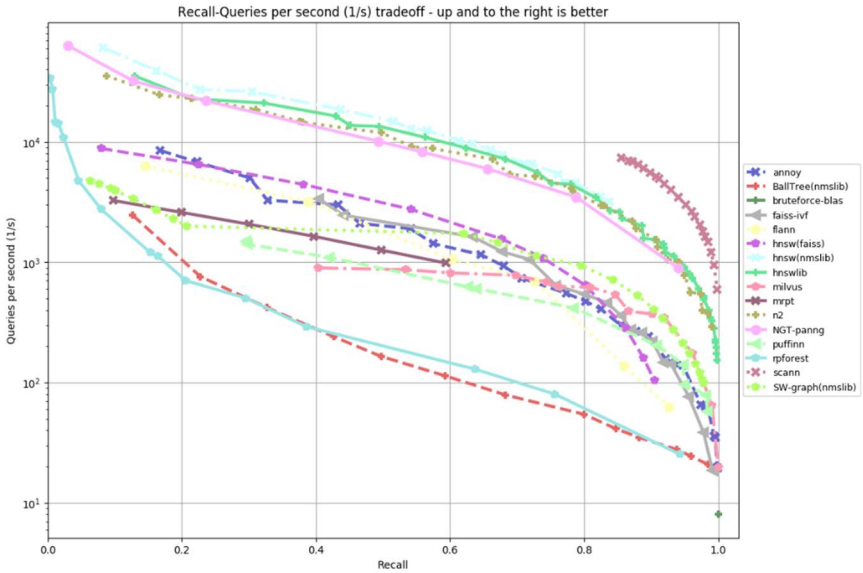
כאשר האיבר הראשון ינסה להביא למינימום וקטורים של אותו אדם, והאיבר השני ינסה להביא למקסימום וקטורים של פרצופים שאינם שייכים לאותו אדם. כיוון שנרצה להימנע מקבלת ערכים שליליים, נוסיף פונקציית מקסימום. בנוסף, ניתן 'להרחיק' תוצאות של פרצופים שונים על ידי הוספת קבוע k, כך שהפרש בין המרחק של פרצופים של אנשים שונים לבין המרחק של פרצופים של אותו איש יהיה לפחות k:

$$L = \sum_{X} max\left(\left\| Y(X^{Ai}) - Y(X^{Aj}) \right\| - \left\| Y(X^{Ai}) - Y(X^{Bj}) \right\| + k, 0 \right)$$

loss כזה נקרא triplet loss, כיוון שיש לו שלושה איברי קלט – שתי תמונות של אותו אדם ואחת של מישהו אחר. כאמור, הפלט של הרשת הנלמדת צריך להיות וקטור המאפיין פנים של אדם, ומטרת הרשת היא למפות פרצופים שונים של אותו אדם לווקטורים דומים עד כמה שניתן, ואילו פרצופים של אנשים שונים יקבלו וקטורים רחוקים זה מזה.

3. מציאת האדם

בשלב הקודם, בו התבצע האימון, יצרנו למעשה מאגר של פרצופים במרחב חדש. כעת כשיגיע פרצוף חדש, כל שנותר זה למפות אותו למרחב החדש, ולחפש במרחב זה את הווקטור הקרוב ביותר ולקטור המייצג את הפנים החדשות. בכדי לעשות זאת ניתן להשתמש בשיטות קלאסיות של machine learning, כמו למשל חיפוש שכן קרוב (כפי שהוסבר בחלק 2.1.3). שיטות אלו יכולות להיות איטיות עבור מאגרים המכילים מיליוני וקטורים, וישנן שיטות חיפוש מהירות יותר (ובדרך כלל המהירות באה על חשבון הדיוק). בעזרת השיטה המובילה כרגע (SCANN) ניתן להגיע לכמה מאות חיפושים שלמים בשנייה (החיפוש ב-100 ממדים מתוך מאגר של 10000 דוגמות).



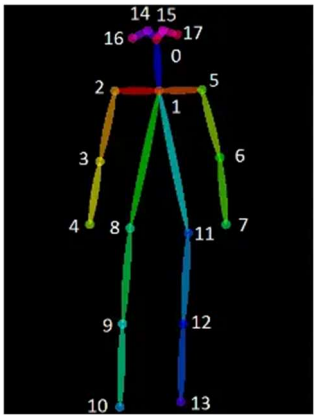
איור 9.9 עבור פרצוף נתון, מחפשים עבורו וקטור תואם בממד החדש המכיל ייצוג וקטורי של הפרצופים הידועים. בכל שיטה יש טרייד-אוף בין מהירות החיפוש לבין הדיוק, ובגרף זה מוצגות שיטות שונות.

מלבד זיהוי וסיווג פנים, יש גם שיטות של מציאת אלמנטים של פנים הכוללות אף, עיניים וכו'. אחת השיטות המקובלות משתמשת בשערוך הצורה של פנים אנושיות, וניסיון למצוא את איברי הפנים לפי הצורה הסטנדרטית. בשיטה זו ראשית מבצעים יישור של הפנים והתאמה לסקאלה אנושית (על פי מרחק בין האיברים השונים בפנים), ולאחר מכן מטילים 68 נקודות מרכזיות על התמונה המיושרת, מתוך ניסיון להתאים בין הצורה הידועה לבין התמונה המבוקשת.

איור 9.10 זיהוי אזורים בפנים של אדם על ידי התאמת פנים לסקאלה אנושית והשוואה למבנה של פנים המכיל 68 נקודות מרכזיות.

9.3.2 Pose Estimation

יישום פופולרי נוסף של אלגוריתמים השייכים לעיבוד תמונה הינו קביעת תנוחה של אדם – האם הוא עומד או יושב, מה התנוחה שלו, באיזה זווית האיברים נמצאים וכו'. ניתן להשתמש בניתוח התנוחה עבור מגוון תחומים – ספורט, פיזיותרפיה, משחקים שונים ועוד. לרוב, תנוחה מיוצגת על ידי המיקומים של חלקי גוף עיקריים כגון ראש, כתפיים, מרפקים וכו'. ישנם כמה סטנדרטים מקובלים, למשל COCO, posenet ועוד. ב-COCO התנוחה מיוצגת בעזרת מערך של 17 נקודות (בדו-מימד):



איור 9.11 מיפוי תנוחה ל-17 נקודות מרכזיות.

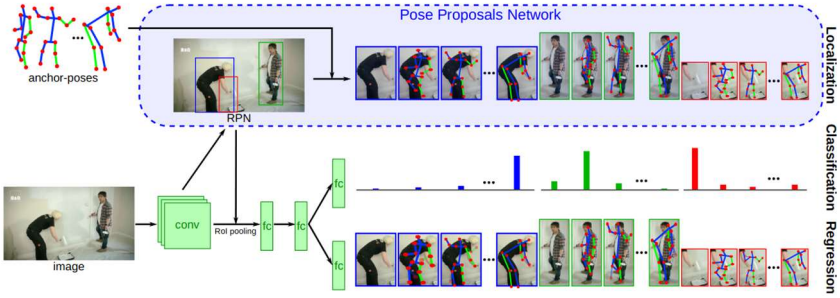
שאר הפורמטים דומים; מוסיפים עוד מידע (למשל מיקום הפה), משתמשים בתלת ממד במקום בדו ממד, משתמשים בסידור אחר וכדו'. כאשר רוצים לאסוף נתונים על מנחי גוף שונים, ניתן לשים חיישנים על אותן נקודות מרכזיות וככה לקבל מידע על מנח הגוף לאורך זמן.

רשת לצורך קביעת תנוחה יכולה לטפל במקרה כללי, בו יש מספר אנשים בתמונה, או במקרה הפרטי של גוף יחיד. המקרה השני כמובן יותר פשוט, מכיוון שישנו פלט יחיד, אותו ניתן לחזות באמצעות רגרסיה (למשל, 34 מספרים שמתארים את המיקומים של 17 הנקודות בפורמט COCO בדו-ממד). במקרה זה ניתן לעשות למידה סטנדרטית לחלוטין של בעיית רגרסיה בעלת 34 יציאות.

ישנן מספר גישות כיצד להכליל את הרשת כך שתוכל לטפל גם במקרה הכללי בו יש יותר מגוף אחד בתמונה. באופן נאיבי ניתן לבצע תהליך מקדים של מציאת כל האנשים בתמונה, ואז להפעיל על כל אחד מהם בנפרד את הרשת שמבצעת רגרסיה, כפי שתואר לעיל. שיטה נוספת פועלת בכיוון הפוך – ראשית כל הרשת מוצאת את כל האיברים בתמונה נתונה, ולאחר מכן משייכת אותם לאנשים שונים. השיטה השנייה נקראת "מלמטה למעלה" (bottom-up), כיוון שקודם כל היא מוצאת את הפרטים ולאחר מכן מכלילה אותם. גישה זו יעילה למקרה בו יש הרבה חפיפה בין האנשים בתמונה, כיוון שאין לה צורך לבצע תהליך מקדים של הפרדת האנשים. השיטה הראשונה, הנקראת "מלמעלה למטה" (top-down), תהיה פשוטה יותר עבור מקרים בהם אין חפיפה בין האנשים בתמונה וכל אחד מהם נמצא באזור שונה בתמונה, כיוון שבמקרה זה אין צורך לשייך איברים לאנשים.

רשת פופולרית עבור משימה זו נקראת Multi-person pose estimation, המורכבת למעשה משתי תת-רשתות ופועלת בשיטת bottom-up. הרשת שאחראית על שיוך חלקי גוף לאדם מסוים, נקראת part affinity fields (PAF), והרעיון שלה הוא לייצג כל איבר כשדה וקטורי. בייצוג זה הווקטורים השונים מצביעים לכיוון איבר הגוף ה'בא בתור' (למשל זרוע מצביעה ליד), וככה ניתן לשייך איברים שונים אחד לשני, ואת כל יחד לגוף מסוים.

רשת פופולרית אחרת, הפועלת בגישת top-down, נקראת LCR-NET, והיא מבוססת על רעיון של 'מיקום-סיווג-רגרסיה' (Localization-Classification-Regression). בשלב הראשון יש תת-רשת המייצרת עוגנים עבור אנשים, כלומר, אזורים בהם הרשת חושבת שנמצא בן אדם, ולאחר מכן הרשת משערכת את התנוחות שלהם. בשלב השני מתבצע clustering לכל העוגנים, כלומר כל עוגן מקבל ציון המייצג את טיב השערוך של העוגן והתנוחה של האדם הנמצא בתוכו. השלב השלישי מלטש את העוגנים ומשקללת את השערוך הסופי בעזרת מיצוע של הרבה עוגנים. שלושת השלבים משתמשים ברשת קונבולוציה משותפת, כמתואר באיור.



איור 9.12 Localization-Classification-Regression.

9. References

Detection:

https://pjreddie.com/darknet/yolo/

https://urialmog.medium.com/

Segmentation:

https://arxiv.org/ftp/arxiv/papers/2007/2007.00047.pdf

Face recognition:

https://docs.opencv.org/master/d2/d42/tutorial_face_landmark_detection_in_an_image.html

http://blog.dlib.net/2014/08/real-time-face-pose-estimation.html