Faster R-CNN

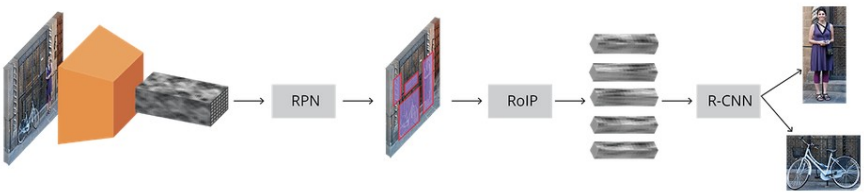
מאמר: https://arxiv.org/pdf/1506.01497.pdf

רקע

Faster R-CNN פורסם ב-2015. Faster R-CNN הוא השידרוג השלישי של R-CNN המקורי.
הכל התחיל ב R-CNN בשנת 2014, שהשתמשו בו באלגוריתם שנקרא חיפוש סלקטיבי כדי להציע אזורים אפשריים המעניינים אותם (שעשויים להיות בהם אובייקטים) ו CNN כדי לשם חילוץ התכונות מאזורים. Fast R-CNN, שפורסם בתחילת 2015, הוא שיפור של R-CNN שם השתמשו בטכניקה שנקראה Region of Interest Pooling ואפשרה לשתף חישובים יקרים והפכה את המודל להרבה יותר מהיר. לבסוף הגיע Faster R-CNN, שם הוצע המודל שנדבר עליו כעת. השוני העיקרי של faster R-CNN מקודמיו הוא שהוא לא משתמש בחיפוש סלקטיבי כדי לייצר הצעת אזורים, אלא ב RPN. עלות הזמן בהפקת הצעת אזורים קטנה בהרבה ב- RPN מאשר בחיפוש סלקטיבי. בנוסף RPN חולק הרבה חישובים עם ה detection network.

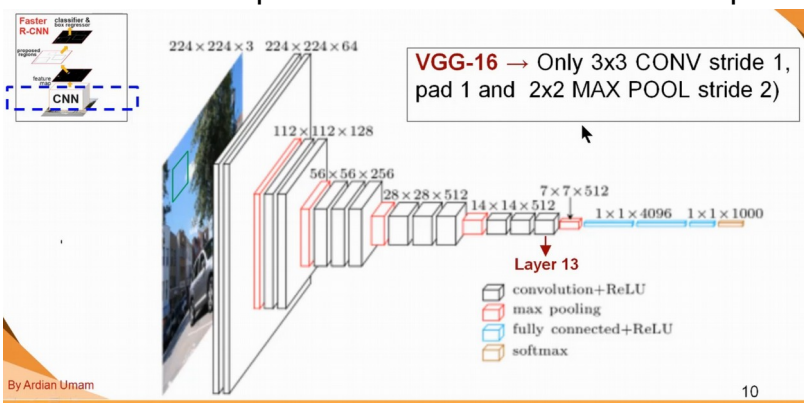
השלבים ב Faster R-CNN

1. שימוש ב CNN שאומן למשימות סיווג ולהשתמש בפלט שלו (feature map) כשכבת ביניים.
2. הפעלת RPN על ה feature map, הפלט הוא אזורים מוצעים.
3. הפעלת RoIP על כל אזור מוצע על מנת לקבל feature map בגודל קבוע.
4. עבור האזורים המוצעים הפעל FC layers שהפלט שלהן הוא class + bbox coordinates.



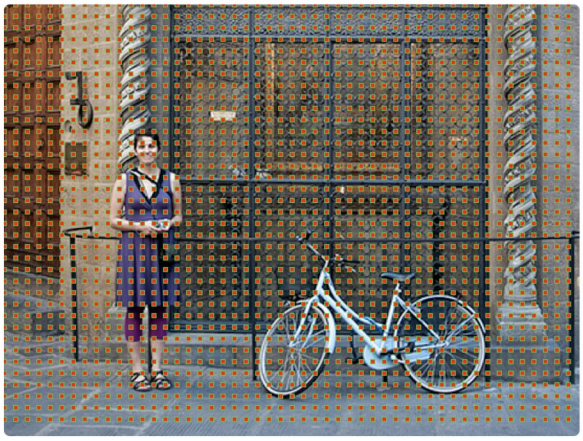
שלב 1:

השלב הראשון הוא שימוש ב- CNN מאומן במשימת סיווג ושימוש בפלט שלו שכבת ביניים.
כותבי המאמר השתמשו ב VGG-16 לביצוע משימה זו.
מתחילים עם תמונת הקלט המוזנת ל VGG. גודל תמונת הקלט מומר ל 600X1000.



ה VGG משמש לשם חילוץ התכונות מתמונת הקלט. התכונות שחולצו באמצעות VGG ישמשו אותנו בשלב הבא. השכבה האחרונה שאנו נשתמש בה בתור ה feature map שלנו היא שכבה

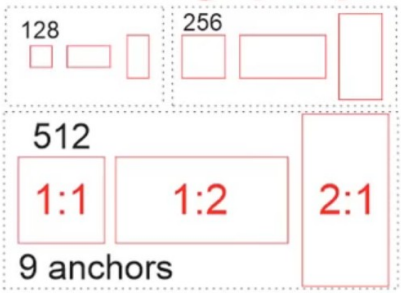
מספר 13. נשים לב כי בשלב זה ה stride הוא 16, כלומר, שני פיקסלים סמוכים ב feature map מתאימים לפיקסלים בתמונת הקלט עם 16 פיקסלים הפרש ביניהם.



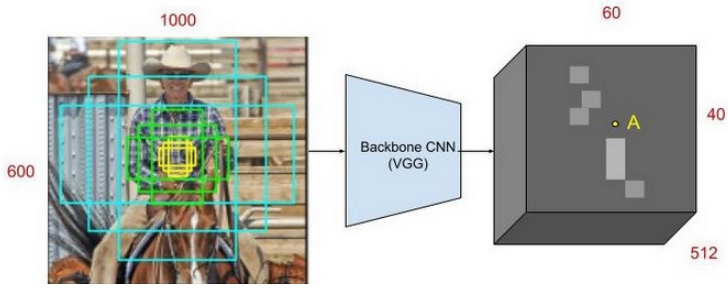
שלב 2:

נשתמש ב feature map מ VGG כקלט עבור ה RPN. עבור כל נקודה ב feature map, עלינו לדעת האם קיים אובייקט בתמונת הקלט במיקום המתאים. זה נעשה זאת על ידי "Anchors" על תמונת הקלט עבור כל נקודה במפת התכונות מה VGG.

כותבי המאמר השתמשו ב anchors ב 3scales ו 3 ratios, כלומר, 9 anchors בסך הכל. עבור כל נקודה ב feature map נריץ את 9 ה anchors הללו.

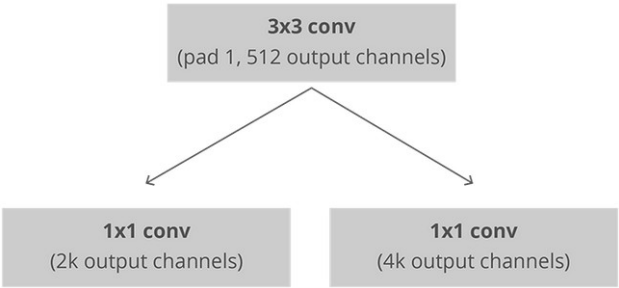


כאשר אנו עוברים על כל נקודה ונקודה במפת התכונות, עלינו לבדוק האם ב-9 ה anchors שלנו המופעלים על תמונת הקלט, יש שם אובייקט או לא. ולשפר את הקואורדינטות של ה anchors כדי שנוכל להעבירם לשלב הבא כאזורים מוצעים בצורה מיטבית.



נשים לב כי גודל ה feature map הוא 60X40 כתותאה מה-stride שהוא כאמור 16 (1000/16X600/16)

ראשית, נפעיל שכבת קונבילוציה עם 512 ערוצים ו גודל קרנל 3X3. ולאחר מכן יש לנו שני שכבות קונבולוציה מקבילות המשתמשות בקרנל 1X1



(K=9 , number of anchors)
קונבולוציה 1X1 אחת עם (2*9)18 ערוצים לשם סיווג (יש או אין אובייקט).
קונבולוציה 1X1 שניה עם(9*4) 36 ערוצים לשם bbox regression (את ההפרש ממרכז ש anchor).

Traning
אז כמה מיקומים אפשריים עבור anchors יש לנו? 40*60 =2,400 מיקומים אפשריים.
על כל מיקום אפשרי אחנו נבדוק עם 9 anchors כלומר 2,400*9 =21,600.
בשלב של האימון נוריד את ה anchors שיוצאים מהשוליים של התמונה, זה ישאיר אותנו עם כ- 6,000 אזורים מוצעים.
נגדיר אזור מוצע חיובי כאזור שה IoU שלהוא הוא לפחות 0.7 עם ה groundtruth, אזור שלילי כאזור שהוא ב IoU הוא מתחת ל 0.3 עם ה groundtruth.
כל mini-batch ב RPN למעשה מורכב מתמונה בודדת, נקח באופן אקראי 256 אזורים מוצעים, כאשר 128 מהם הם חיוביים ו 128 מהם שליליים.

Loss
פונקציית ההפסד שלנו מורכבת משני חלקים:

$$L(p_i, t_i) = \underbrace{\frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*)}_{\text{object/not object classifier}} + \lambda \underbrace{\frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)}_{\text{box regressor}}$$

חלק ראשון:

$$L(p_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*)$$

p_i : predicted probability

$p_i^* \begin{cases} 1 \text{ for pos anchor} \\ 0 \text{ for neg anchor} \end{cases}$

N_{cls} : numbers of anchors in minibatch (512)

cross-entropy loss

החלק השני:

$$L(\quad t_i\ ) = \qquad\qquad +\lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*)$$

λ = constant value

N_{reg} = number of total anchors

$p_i^* \begin{cases} \text{1 for pos anchor} \\ \text{0 for neg anchor} \end{cases}$

$$L_{reg} = \text{smooth}_{L_1}(t_i - t_i^*) \quad \text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

t_i = predicted box; t_i^* = ground truth box

$$\boxed{\begin{aligned} t_x &= (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a, \\ t_w &= \log(w/w_a), \quad t_h = \log(h/h_a), \\ t_x^* &= (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a, \\ t_w^* &= \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a), \end{aligned}}$$

אז החלק הזה רלוונטי אם יש אובייקט באזור המוצע.

Test

אז כמה מיקומים אפשריים עבור anchors יש לנו ? 40*60 =2,400 מיקומים אפשריים.

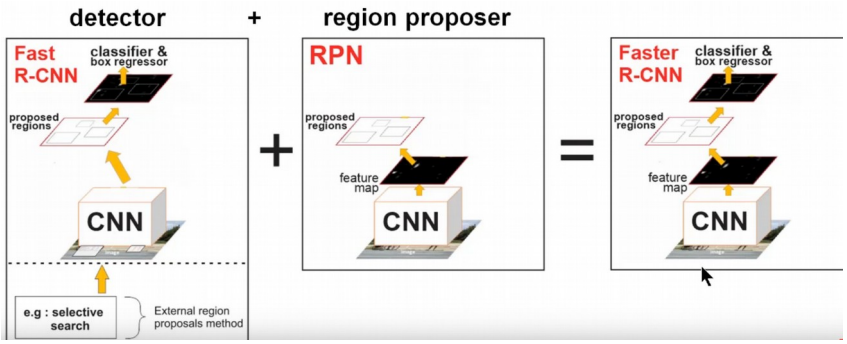
על כל מיקום אפשרי אחנו נבדוק עם 9 anchors כלומר 2,400*9 =21,600.

זה יותר מידי, יאט ממש את הרשת.

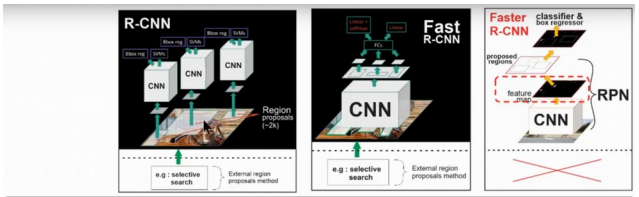
כותבי המאמר הציעו על מנת להוריד מספר ה anchors העצום שקיבלנו לבצע את הפעולות הבאות:

- להוריד את ה anchors שיוצאים מהשוליים של התמונה.
- נפעיל על האזורים המוצעים שנותרו NMS עם ערך סף של 0.7.

לאחר מכן נשאר עם כ 2,000 אזורים מוצעים וזה משהו שנוכל להתמודד איתו.



כעת נעביר את 2,000 האזורים המוצעים מה RPN אל ה Fast R-CNN detector.



	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image	50 seconds	2 seconds	0.2 seconds
Speed-up	1x	25x	250x
mAP (VOC 2007)	66.0%	66.9%	66.9%

* Standford lecture notes on CNN by Fei Fei Li and Andrej Karpathy