

סדרה בלמידה עמוקה – סיכום על פि המאסטרו

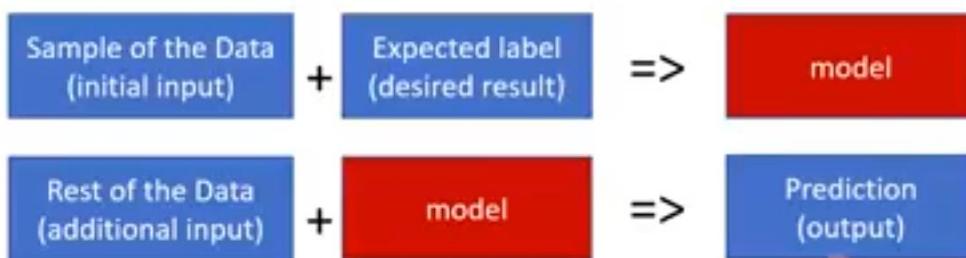
הרצאה 1:

מאשין לארנינג (ML) – למידה מוגדרת בתור שיפור ביצועים בתור פונקציה של ניסיון (הלברט אלכסנדר סימן – במאה הקודמת).

למ' היא תוכנה במחשב שבמקומם שנרשום כיצד להגיב לקלט אפשרי אנו רוצים שהוא תלמיד על פי ניסיון כיצד להגיב לאוטו קלט בעזרת אוסף של נתונים.

הweeney הבסיסי בLM שנרצה להבהיר נתונים בתוספת איזשהו תוצאה רצiosa והתוצר של שני הדברים הללו יחד עם הקודד יהיה מודול ולא יהיה הפלט הרצוי, כשמייקח את המודול ביחיד עם שאר הנתונים שאין לנו עליהם את התוצאה הרצiosa, המטרה היא שנתקבל פרדיקציה והפרדיקציה הזאת בשאיפה תהיה מוכללת ככל שאפשר.

Machine Learning applies the statistical inference method and perception to algorithms i.e. predict by learning from a previously seen sample of the data



אנחנו לא רוצים שההתוצאות שלנו יהיו בעלי ערך רק על הנתונים שעיליהם למדנו, (שינון) אנחנו גם לא רוצים שההתוצאות יהיו מוכללים מדי וכך לא מספיק טובים (הפשטה יתר), אנחנו נרצה שהם יהיו מספיק מותאמים לנ נתונים שחשפנו אליהם בשלב בניית המודול בשביבלי ליציר תוצאות בעלות ערך אך לא מותאמות יתר על המידה בשביבלי לא ליציר אלמנט של שינון אלא באמצעות מילון מילוי.

אננו מחלקים את העולם של ה-ML לארבעה תת-תחומים עיקריים:

Supervised (למידה מונחה) – יהיה רוב מה שנדבר עליו בקורס, למידה שההתוצאה שלנו ידועה מראש, לדוג' אנחנו ננסה לחזות את הטמפרטורה שתיהיה מחר אבל לפחות עבור הנתונים ההיסטוריים בהינתן איזשהו נקודה בזמן אנחנו יודעים מה תהיה הטמפרטורה גם יומם לאחרת של אותה נקודה. דוג' נוספת בהינתן תמונה, נרצה להגיד מה האובייקט שנמצא במרכז התמונה אבל יש לנו עבור דאטא סט תיווג מה יש באזורה תמונה.

Unsupervised – אנחנו מייצרים איזשהו תהליך שאנו לא יודעים מראש איזה חלוקה תיווצר ממנו, בסוג זה של בעיות אנחנו יכולים למצוא בעיות מסווג סיגמנטציה או קלואסטרינג (clustering) קבוצות שיש בינהן משותף בתוך הנתונים שאנחנו לו דוקא יודעים להגיד אותם מראש. לדוג' בוא נדמיין שאנו לא יודעים מה ההבדל בין נורות לבין שלוחנות, אז אם נקבל נתונים שמכילים את שני המוצרים האלה, בעזרה יכולת זיהוי צדאת או אחרת מבל' לדעת אפילו מה הם שני המוצרים האלה נוכל להפריד ביניהם לשתי קטגוריות שונות (צורה, גודל, משקל, חומר וכו').

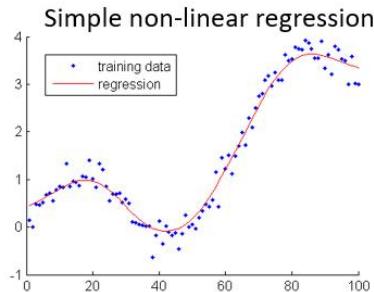
Semi-supervised – נובעת מהעובדה שיש לנו הרבה מקרים גם פרקטיקה וגם במחקר של יציר דוגמאות מתייגות זה תהליך שהוא מאד יקר גם במשאבים וגם בכסף, ולפעמים פשוט אין מספיק תיעדים של דוגמאות מתייגות (פשוט לא ניתן להשיג כללה כמו מחלות נדירות שגם אם היינו רוצים להשיג עוד דוגמאות כללה אז פשוט אין) ופה יש trade-off בין כמה אנחנו רוצים להשיקע בתהליך תיוג הנתונים לבין התוצר שלנו תהיה טובה או מדוקית ואז נוצרה גישה שהיא משלבת גם חלקים שהם Unsupervised בעצם של מידיה של מאפיינים שהם רוחניים בנתונים או קיבוץ של הנתונים לצד מידע של נתונים מתייגים ואז תהליך של תיוג מתמשך, אנחנו מתייגים כמות קטנה של נתונים ואנחנו מפעילים אלגוריתם כדי ליציר המלצות תיוג של נתונים נוספים ואז בודקים אותנו המלצות תיוג ולאחר מכן מאמנים מחדש על אותם נתונים שבחנו. כמובן שליציר המלצת תיוג לוקח הרבה הרבה זמן ואז אנחנו רואים מצב זה שבתהליך אינטגרטיבי אנחנו יוצרים עוד ועוד דוגמאות מתייגות אבל התהליך עצמו מצריך שימוש במודלים עבור אותו תיוג.

Reinforcement – לא ממש נתעסק בקורס* היא כן הולכת וצוברת עניין וייתר שימושית, ומה שמיוחד בסגנון בתחום היזטורי, שיש לנו תיוג אבל לא עבור כל דוגמא בנתונים, אלא תיוג שנמצא עבור רצף של מספר דוגמאות. לדוגמא רצף של מHALCs במשחק שאנחנו יודעים מה התוצאה שהגענו אליה לאחר מספר צעדים ולא אחריו צעד אחד בלבד (בקשר של שחמט, לפעמים אנחנו עושים צעד שנראה לא הגיוני או אפילו מפסידים כל' בצעד מסוים בשביל שבעזרת מספר נוספים של צעדים נקבל יתרון בזכות תחילת המהלך).

<ul style="list-style-type: none"> • Supervised • Unsupervised • Semi-supervised • Reinforcement 	<p>learn with a known target e.g: classify pictures to given categories, predict regression results</p> <p>learn without a known target e.g: Split data to homogeneous groups (aka segmentation or clustering) Find intrinsic structure within the data</p> <p>Hybrid approach: Since most of available data is unlabeled, try improve our predictions by using lots of unlabeled data with small amount of labeled data</p> <p>Relatively new approach: Learn multi-stage problems without a known target for each step but rather a reward for the whole process</p>
--	--

דוגמאות של Supervised

- לשערר ערך של טרניזקציה.
- בעיות של Time series prediction.
- השתיכות של קבוצה צאת או אחרת (קלסיפיקציה).
- ניתוח תמונה, מتوزר הרבה קטגוריות שונות לסוג או לשיר תמונה לפי קטגוריה מتوزר מגוון רחב של קטגוריות.



Time series regression prediction

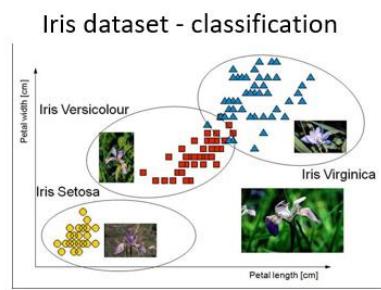
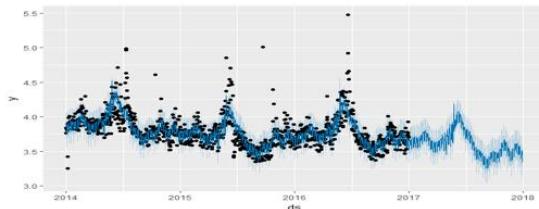
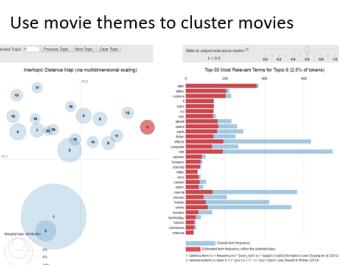


Image-net dataset large scale classification

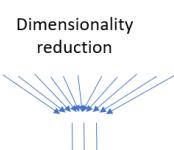
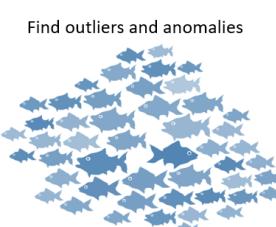


דוגמאות של Unsupervised

- לסוג אוכלוסייה לקבוצות שיש מכנה משותף (דוג' חברות סלולר יכולות לנסוטו לסוג לקוחות לפי התנהגות דומה ואז לנסוטם למכור להם מוצרים או מוצריהם דומים שהם חשובים שיעניין). אולם לפ' אותו סיווג).
- לפי ניתוח של מופעים ושכיחות ללא תיוגים, ניצ'ר קבוצות שונות (דוג' לפ' שם של סרט לתייג את סוג הסרט).
- זיהוי של אונומליות, למצוא נתון שמתנהג שונה מתנאים אחרים ולא שייר ולא זהה לשאר הנתונים. יש מספר לא מבוטל של אלגוריתמים שמתיחסים לתנום זהה.

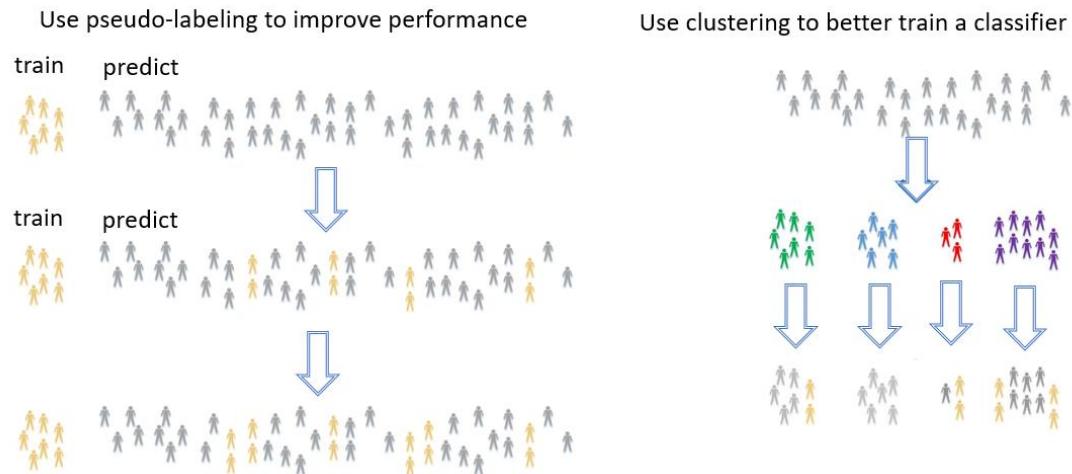


Extract characteristics to segment the population



דוגמאות של Semi-supervised

- אפשר לנסות למשל להסתכל על סגמנטים שונים בתור איזשהו מאפיין נוסף לביעית קולסיפיקציה, למשל אפשר להשתמש במודל כדי ליצור תיוג לדוגמאות חדשות (או המלצה תיוג) ובעזרת pseudo-labelling בעצם אנחנו יכולים להשתמש בתוצאה של מודל בשביל לשערך מה תהיה תוצאה של הליבל של חלק אחר באוכלוסייה שאנו לא יודעים מה התיאוג האמתי שלו ואחר כך לאמן באמצעות כל האוכלוסייה את הגרסה הבאה של הפרדיקציה.



דוגמאות של Reinforcement

- ככל שהוא נחשפים ליותר ויותר דוגמאות אנו ללא הצלחה והמודל שלנו עדין לא שולט, אך ככל שהוא נחשף ליותר ניסיונות והוא יותר מקרים המודל לומד כיצד לשЛОוט בסיטואציה (נאמר בהקשר של המושך של Atari).

Atari game learned with deep Q learning



Multi armed bandits

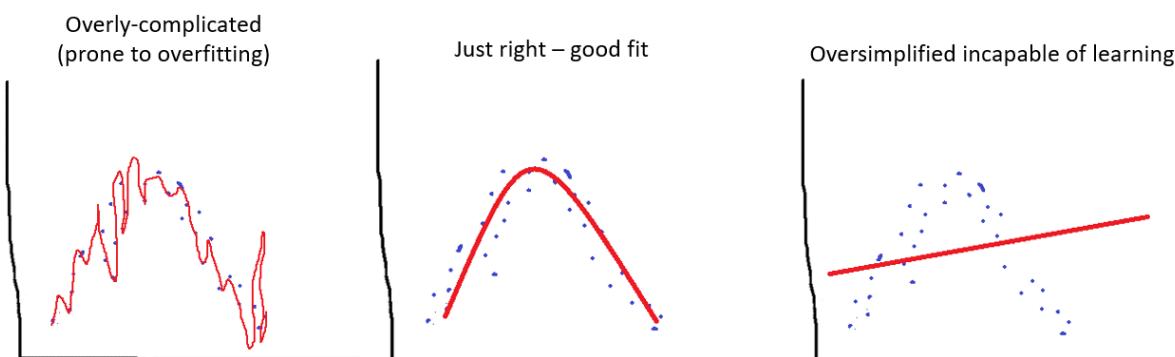
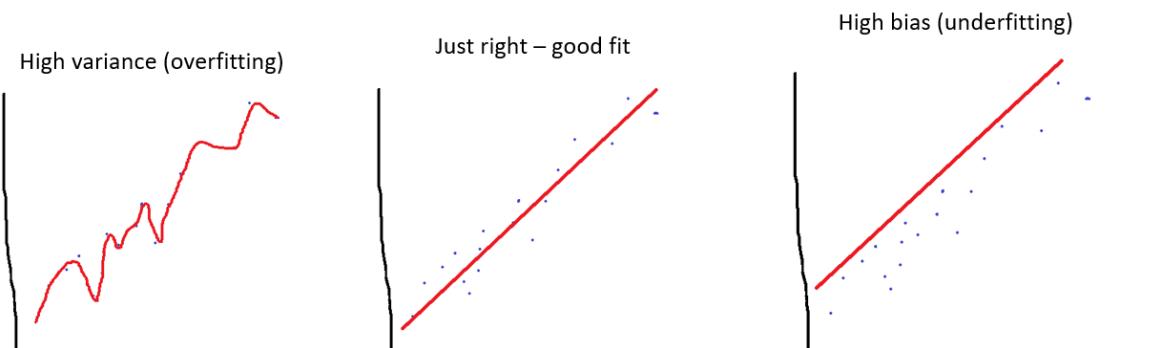
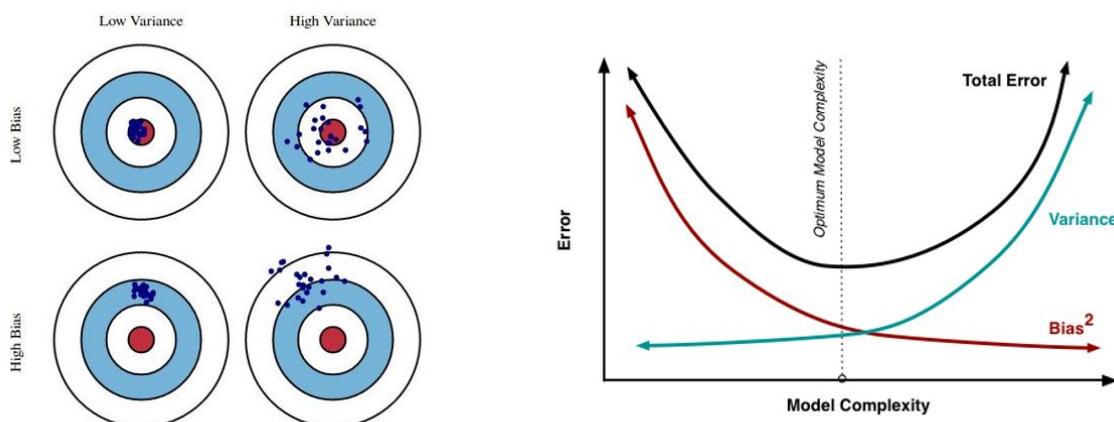


מושגים בסיסיים בעולמות ה-ML:

Bias vs. Variance – the overfitting problem

מה שאנו נשרף זה להגעה ל-Variance-Bias (התאמת טובה). בעצם כשאנו מדברים על עיקרון ה- Bias vs. Variance אנחנו מדברים על הטעות של המודל שלנו כפונקציה של מורכבות המודל, אנחנו נראה בתחילת כשהמודל מאוד פשוט אז הטעות שלו תהיה מאוד גדולה והטעות נובעת בעיקר מ-Bias. השונות שלו (Variance) תהיה נמוכה, המודל לא מאוד מורכב ולכן לא מבדיל בין אלמנטים שונים.

כל לנו מגדילים את מורכבות המודל אנחנו מגיעים לסוג של ירידת Bias-Shallala (לא מקבילה לאותו היוף עלייה של Variance). וזה יש את נקודת שאננו מגיעים לאופטימום של מורכבות המודל ומאותה נקודת אנחנו מעלים את השונות ולמרות שה-Bias ממשיך לירידת (בקצת) אותה עלייה של שונות גורמת לשגיאה הכללית במודל.



:Validation

עכשווי שהבנו שיש לנו אלמנט שהמודל יכול לשנות, אז אנחנו מבינים שבמהלך האימון אנחנו צריכים ליצור רפארנס שהוא לא מהנתונים שעליים אנו לומדים או אליו הם אנוחושפים את המודל, אלא נתונים שאנו נבחר בשבייל לבדוק את יכולת הכלכלה של המודל המאומן.

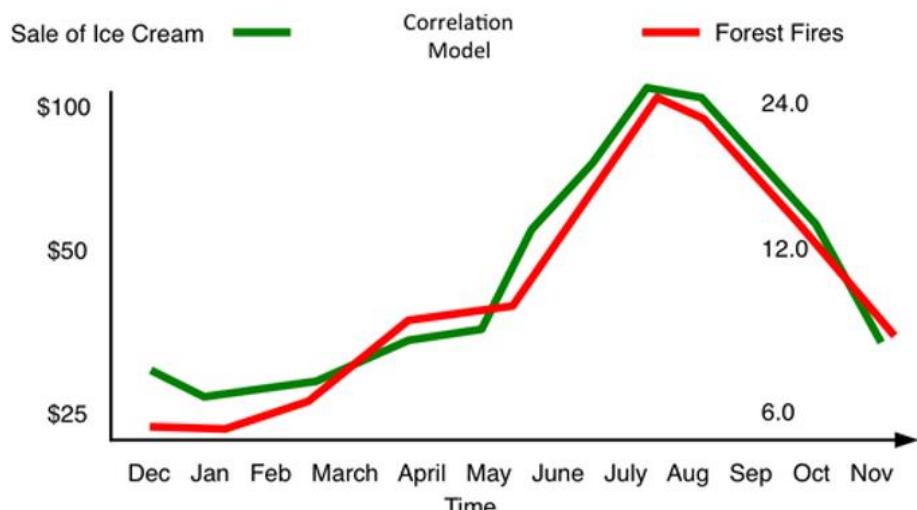
שימושים שעבורם משתמש בвалиדציה הינן:

- כדי לבחור את ה-Hyper-parameters של המודל.
- איזה מודל מתאים לבעה שלנו (מודל יותר פשוט/יותר מורכב)
- Early stopping – לעיתים אנחנו רואים שהמשר למודל יוצר שינוי ולא יוצר יותר הכללה אז עדיף לעצור את תהליכי הלמידה/אימון.
- Train-test split
- K-fold
- (loo) Leave one out – מקרה פרטי של K-fold-K כאשר אנחנו מאמנים על כלל הנתונים בלבד דוגמא בודדת ואז בודקים את עצמנו על אותה דוגמא בודדת. (leave P out) – אותו דבר רק שימושים אחד בחוץ).
- Group K-fold
- Leave one group out.
- Time series – כל מה שקשר לSD שיש תלות בזמן, אנו לא יכולים להניח שאימון על דוגמאות שנבחרו בזמן רנדומלי ייבנו תוצאות כמו אימון על נתונים שנבחרו על פי זמן.
- Unbalanced data – כישש לנו נתונים מועטים כלומר 99% אחוז מהבדיקות עם ערך שלילי ורק 1% עם ערך חיובי אנחנו עלולים להגיד למצב שאנו נdagום בצורה צאתת שלא תהיה לנו ייצוג בכלל של אותן רשומות עם ערך חיובי, או שנגיעה למצב שיש הטיה למודל שלנו.

Causality vs. Correlation

הרבבה פעמים אנו רואים קורלציה ואני מניחים שיש קשר של סיבה ותוצאה זהה כਮובן לא תמיד המצב, לעיתים יש גורם שלישי שגורם לשניהם או לקורלציה בין שניהם ולפעמים זה פשוט מקרים.

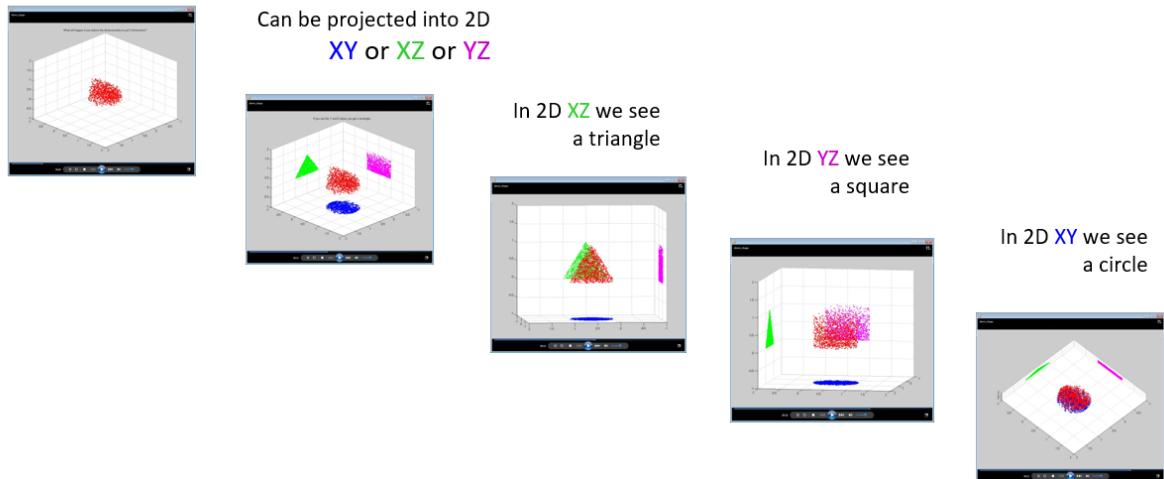
בדוג' יש מכירות של גלידה ושריפות של יערות (שבורר שכנראה אין באמת קשר בין שני הדברים).



Dimensionality reduction

- בתמונה הראשונה (שמאל למעלה) אנו רואים מצבור של נקודות במרחב 3D.
- בתמונה הבאה ניתן לראות את התלה שיש לאותן נקודות בכל אחד מהממדים.
- בכלל אחת משאר התמונות ניתן לראות את אותה התלה בממדים השונים.
- כמובן שלראות התלה מסוימת ואת הצורה זה נחמד אבל מה שניתן ללמידה מזה כuish לנו הרבה כמו 100 ממדים או יותר ואנו מייצרים איזשהו תלתה על מרחב קטן יותר והאם אותן תופעות שאנו מבחינים בהן באותו ממד מצומצם הם אכן התופעה שאנו אמרו לפוטה בה או רק אלמנט/ביטוי של אותה הורדת ממד.

A cloud of points in 3D



Garbage in garbage out

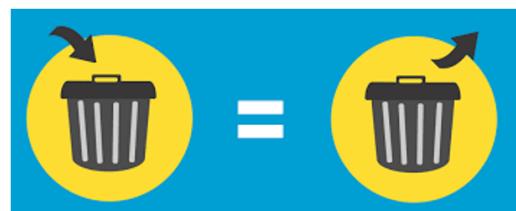
איך הנטונים, איךות התיאוג אם נזין את המודל עם דוגמאות עם הרבה רעש ומעט סיגナル אז יש אפשרות גדולה שנסוווג את הרעש במקום את הסיגナル הרצוי.

התיאוג אנושי יש טעויות אנווש

תיאוג לא מדויק או נכון עקב חוסר משבבים או רצון של מישחו לסייע את העבודה מהר. מה שיקירה שאז נחשוף את המודל לנוטונים לא מדויקים שהם הוא ילמד.

דבר נוסף שיכול לקרות זה שאחננו משתמשים באיזשהו כל מדידה עם סטייה (לדוג' מד משקל עם סטיית משקל) התוצאות שנקבל יהיו בעלי סטייה ואחננו נctrkr לדעת זאת על מנת שתאת התיאוג הנכון או הערך הרצוי חיב להיות בהתאם לסטייה הזאת כי אם לא אז ברגע שנשתמש בכל מדידה ללא סטייה נקבל תוצאות שונות מאשר שהניב המודל.

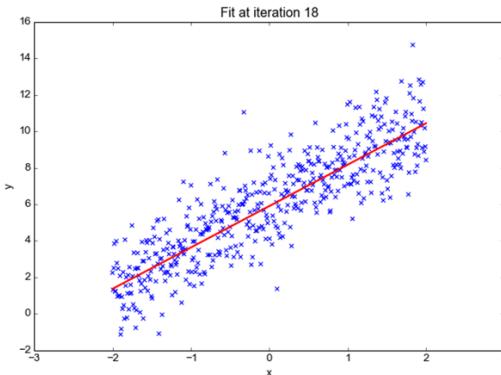
- Manual labels
- Badly calibrated meters
- Noisy data source



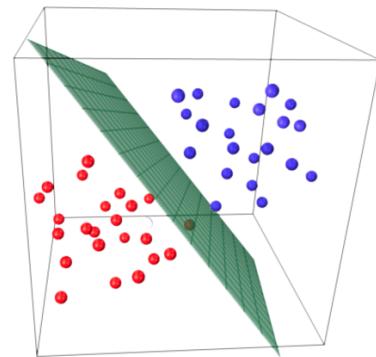
משפחות נפוצות של מודלים בסיסיים:

- **מודלים לינאריים** – בעיות רגסיה, בעולם של קלאסיפיקציה אנו נרצה לסוג נקודה לקבוצה מסוימת, קבוצה א' או קבוצה ב', כמובן שבעיה זאת לפעמים יכולה להיות פשוטה אבל גם מאוד מסובכת.

Linear regression

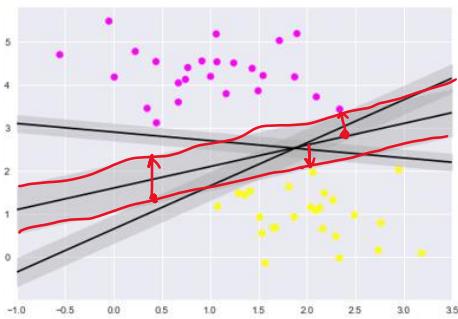


logistic regression

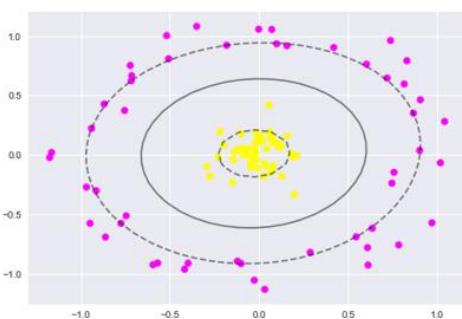
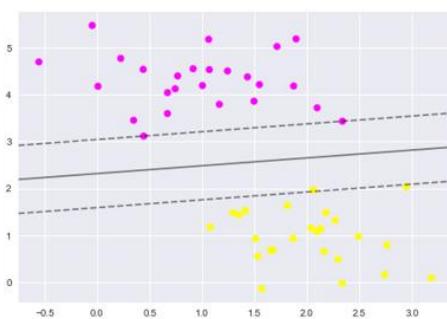
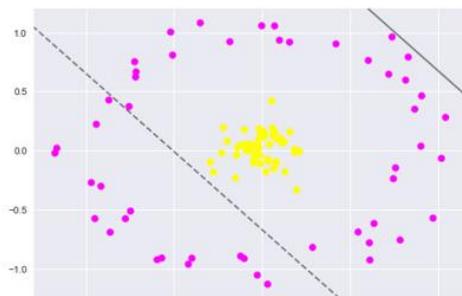


- **Support Vector Machines (SVM)** – הרעיון כאן שיכולים להיות לנו הרבה קוויים מפרידים (כמו בדוג' השמאליות העילונה), השאלה איזה מבין הקווים יכולת לתאר את ההפרדה בצורה הטובה ביותר. משפחה של SVM משתמשת לסוג אלגוריתמיים שנקראיםHighest margin, כלומר אנו יכולים לחת את הנקודה הכי קרובה לגבול ההפרדה מבין כל אחת מהקבוצות ולומר שאם המרחק הוא האגדול ביותר אז אנו נבחר אליו קו מפריד שבו גבול ההפרדה יוצר חור gap גבוה ככל האפשר. כמובן שבגבול ההפרדה לא חייב להיות לינארי ונימtan לראות דוגמא לגבול הפרדה שהוא רדייאלי.

SVM (linear kernel)



SVM (rbf kernel)



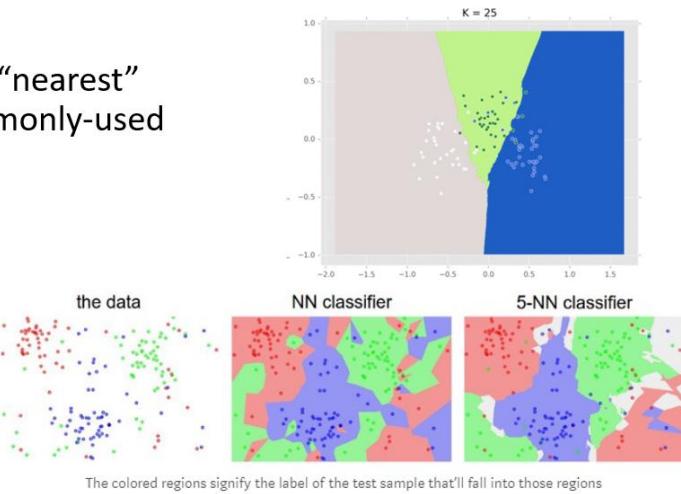
- **קלסיפיקציה מבוססת מרחק (KNN)** – כאן אנו רואים שהחלוקת שלנו היא לאו דווקא לינארית, אלא מבוססת על קרבה באוטו ממד של המשתנים המסבירים שלנו. המרחק לא חייב מחושב לפי מרחק אוקלידי (למרות שבמקרים רבים זה נהוג). צריך לזכור שיש מקרים שפיטוט לא ניתן לחשב מרחק אוקלידי ואנו נרצה להשתמש בשיטה נג'יד במרחב מנהטן (כפי שאפשר לחזור במנהטן דרך הבניינים וצריך לעבור מרחק בצורה של הבלוקים של הרחובות וזה המרחק הנכון עבור אותה עייה כי למרחק אוקלידי אין משמעות אמיתית).

עבור בחירת K נמור אנו עלולים לקבל לפעמים מצב של Overfitted או פחות מוכל, ככל שנעלת את K אז גבול ההפרדה הופך להיות יותר "חלק".

- For each point in validation / test set predict the average of the k nearest points:

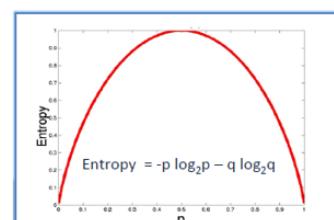
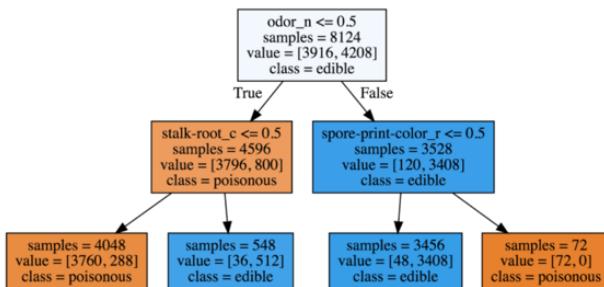
- Distance metric used to define “nearest” may vary. These are some commonly-used distance metrics:

Camberra :	$d(x, y) = \sum_{i=1}^m \frac{ x_i - y_i }{ x_i + y_i }$	Euclidean :	$d(x, y) = \sqrt{\sum_{i=1}^m (x_i - y_i)^2}$
Minkowsky :	$d(x, y) = \left(\sum_{i=1}^m x_i - y_i ^p \right)^{1/p}$	Manhattan / city - block :	$d(x, y) = \sum_{i=1}^m x_i - y_i $
Chebychev :	$d(x, y) = \max_{i=1}^m x_i - y_i $		



- **עצי החלטה** – אם יש לנו במצב ההתחלתי 8,000 דוגמאות עבור בעיית קלסיפיקציה אנו נרצה לייצר פיצולים עבור אותן דוגמאות באמצעות המשתנים המסבירים על מנת להוריד את ממד האי-סדר (אנטropיה). אנו נרצה לבדוק כל משתנה חיתוך אפשרי שאם נבחר בו אנו נקבל את הפחתה המשמעותית ביותר של האנטropיה.

בתהליכי של עץ הבניה שלו היא במבנה באופן דטרמיניסטי, אך אנו נרצה ליצור מגוון עצים שם שונות מסויימת שהשלוב שלהם ייתן תוצאות שאן יותר טובות מכל עץ באופן פרטני.



- Bagging – בתהליך של עץ הבניה שלו היא בניה באופן דטרמיניסטי, לכן אנו נרצה ליצור מגוון עצים עם שונות מסוימת שהשילוב שלהם ייתן תוצאות שהן יותר טובות מכל עץ באופן פרטני. לכן נשתמש בשיטה שנקראת Bagging (כמו אצל עמיאל האחד והיחיד).



- Boosting – נשתמש באלגוריתם חלש וכל פעם נשתמש באותו מקרוב על מנת לשפר את התוצאה, בפועל מה שנעשה שעבור כל תוצאה ננסה ללמידה את הפרש בין התוצאה שלנו לתוצאה האמת ובלכל פעם המודול שלנו ינסה ללמידה יותר טוב את התוצאה האמיתית ובשאיפה שנגיעה לתוצאה מדויקת או מספיק טובה ביחס לתוצאה האמת. (סיפר את סיפור הגOLF שוב).

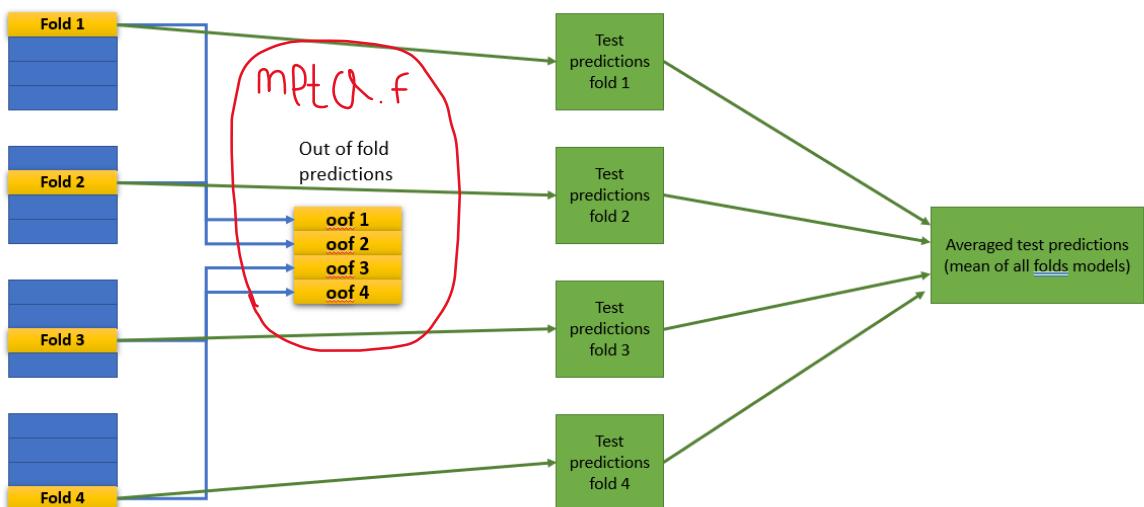
- **Boosting** – train an iterative set of “weak” models and re-weight samples to make each successive model try to correct the previous model’s errors



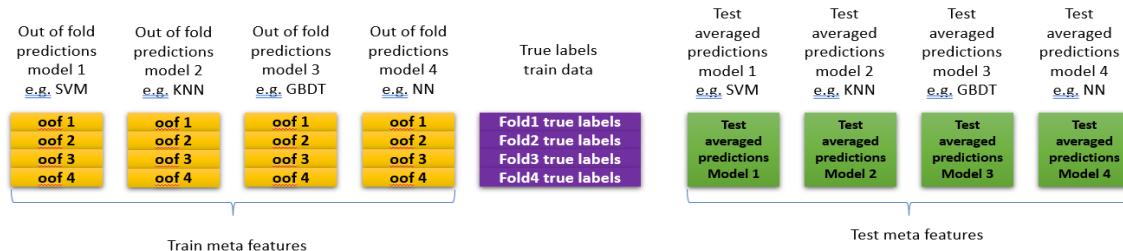
שיטות אנסמבל – ראיינו כרגע שתי שיטות אנסמבל, בסופו של דבר יש מספר דרכים לקבוע כיצד לשקלל אותם, מה שחייב לזכור שהמודלים צריכים להיות בלתי תלויים אחד בשני ובנוסף להם יהיו יותר טובים מນיחוש רנדומלי.

- simple average $(\text{Pred1} + \text{Pred2} + \text{Pred3})/3$
 - Weighted average $0.2 * \text{Pred1} + 0.3 * \text{Pred2} + 0.5 * \text{Pred3}$
 - Majority vote Mode ($\text{Pred1}, \text{Pred2}, \text{Pred3}$)
 - stacking Multi-Level model ensemble

– מצב שבו אנחנו נאמן את המודל שלנו על מספר folds שונים ובכל פעם נשתמש במודל אחר על מנת לקבל סיגו או תוצאה, לאחר מכן נוכל ללקח את כל התוצאות שקיבלנו (Out of fold predictions) ולומר פרדיקציות שלא מבוססות על הנתונים שהתאמנו עליהם עבור כל נתוני האימון שלנו, ואז נוכל לראות בה כפיצר חדש או בשם המקובל meta features.



ונכל לעשות את זה מספר רב של פעמים עם מודלים שונים ואז לאמן מודל חדש עם הפרדיקציות החדשות.

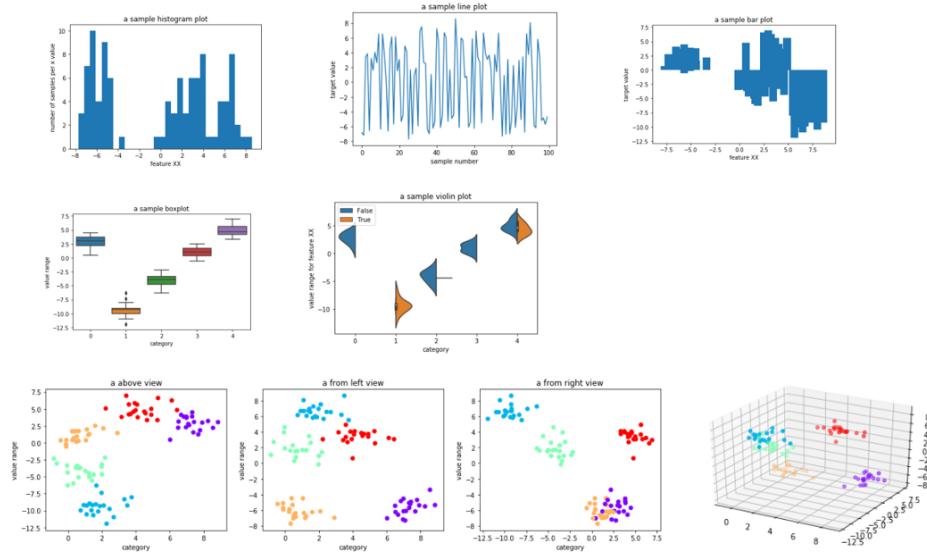


After training several models using this method (4 different models in this example) We can now train a new model using our newly formed meta features

* Note that we can either train our meta model using only these new features or use the new features along with our original train data for training

- Plots – יש לנו סוגים שונים של plots שב邹רתם ניתן לקבל תובנות שונות על הנתונים שלנו, לדוג' התפלגות/ שירט קבוצות שונות/ טווח ערכים.

- Histograms
- Bar plots
- Line plots
- Box plots
- Violin plots
- Scatter plots
- Heatmaps
- Multiple plots



הרצאה 2:

בשיעור זהה נראה כיצד אפשר לייצר פתרונות מבוססים רשותות נירוניים, אנחנו נבחן את השימוש ברשותות ליזיהו ספורות בכתב יד.

ניתן להסכל על רשות כל בניין שאנו מכנים לו קלט והוא עובר בכל הרמות עד שהוא מגיע לסוף הבניין או כמו על תהליך כימי ובשאיפה אם התהליך הוא טוב אז אותו פלט יהיה קרוב לפלט הרצוי.

דבר על:

Layer (type)	Output Shape	Param #
<hr/>		
conv2d_23 (Conv2D)	(None, 26, 26, 32)	320
conv2d_24 (Conv2D)	(None, 24, 24, 32)	9248
dropout_7 (Dropout)	(None, 24, 24, 32)	0
max_pooling2d_9 (MaxPooling2D)	(None, 12, 12, 32)	0
conv2d_25 (Conv2D)	(None, 10, 10, 16)	4624
conv2d_26 (Conv2D)	(None, 8, 8, 16)	2320
dropout_8 (Dropout)	(None, 8, 8, 16)	0
max_pooling2d_10 (MaxPooling2D)	(None, 4, 4, 16)	0
flatten_6 (Flatten)	(None, 256)	0
dense_6 (Dense)	(None, 10)	2570
<hr/>		
Total params:	19,082	
Trainable params:	19,082	
Non-trainable params:	0	

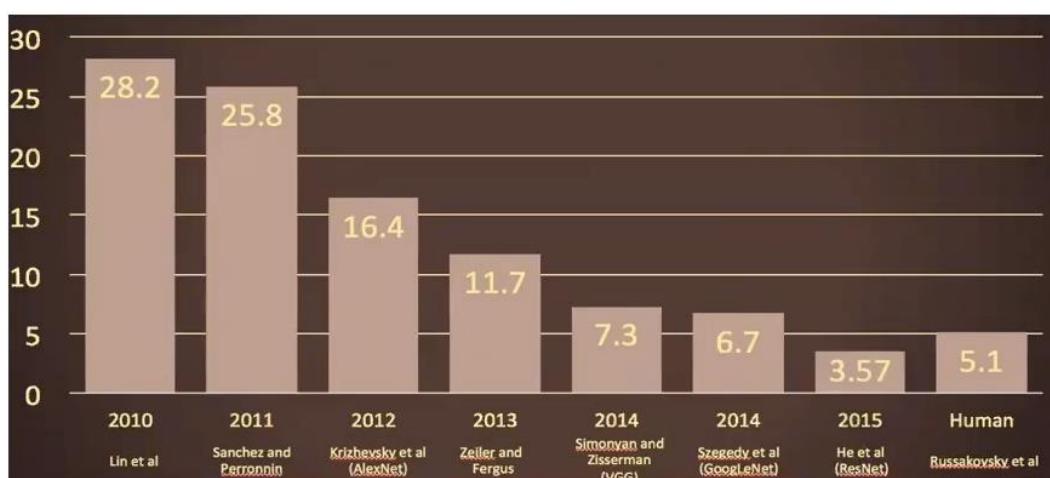
- שכבות
- פונקציות הפסד (loss)
- פונקציות אקטיבציה
- אופטימיזרים

במשך נtabון גם על החשיבות של pre-processing בתהליך עצמו.

קצת בסיס על רשותות נירוניים:

השימוש ברשותות נירוניים התחל כבר בשנות ה-40 של המאה הקודמת, עצם ההשראה הגיינית מההבנה שהמוח של חיוט יש נירונים שיורם בהינתן סף מסוים שהמטען שלהם עבר, ובעצם הצורך של ריבוי של אותם נירונים שפועלים ביחד אחד עם השני, אנחנו מגיעים למערת ייצוג של הרבה מאוד מצבים רצויים. התחום נחassoc לתיאורטי לתקופה מאד ארוכה בעיקר בגל חוסר משאבים והאי יכולת של כוח חישוב.

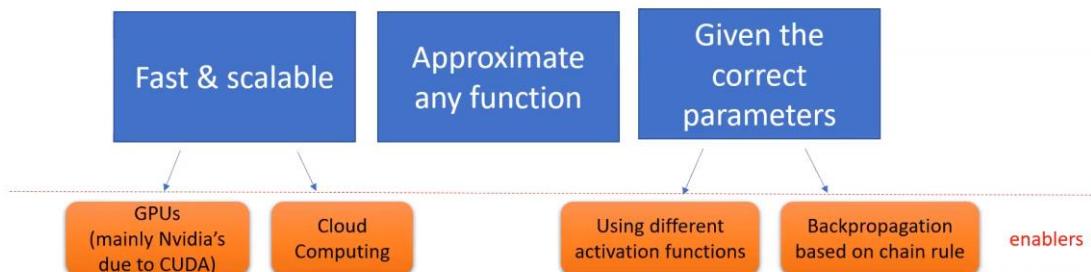
ב-2011 בחור בשם אלכס קירשבקי (שנחשב לאחד האבות המייסדים של התיאוריה המודרנית), לפק בעצם אלגוריתם שהוא ידוע תקופה יחסית ארוכה לפני כן ויצא יישום שלו לתחום שנקראת ImageNet אותו יישום של רשות נירוניים מוכוונת מטרה של ניתוח תמונה יצר שיפור מאד משמעותי באותה תקופה.



הסיבה שדווקא בעשור האחרון התחום זהה חדל מלהיות תיאורתי בלבד וכן רואים בו שינוי ופריצת דרך, נובע מספר גומרים:

- צריך יכולת שהפתרון הזה יעבור מאד מהר ובצורה סקלרית.
- אנחנו נרצה שתיהיה לנו יכולה לשערך כל סוג פונקציה.
- שתיהיה לנו יכולה ליצור תהליך שמתכנס לאוטו פתרון (לא נרצה לסרוק את כל המרחב החיפוש האפשר בשבייל להגעה לפתרון זהה).

- Universal approximation theorem
a NN with a single hidden layer can:

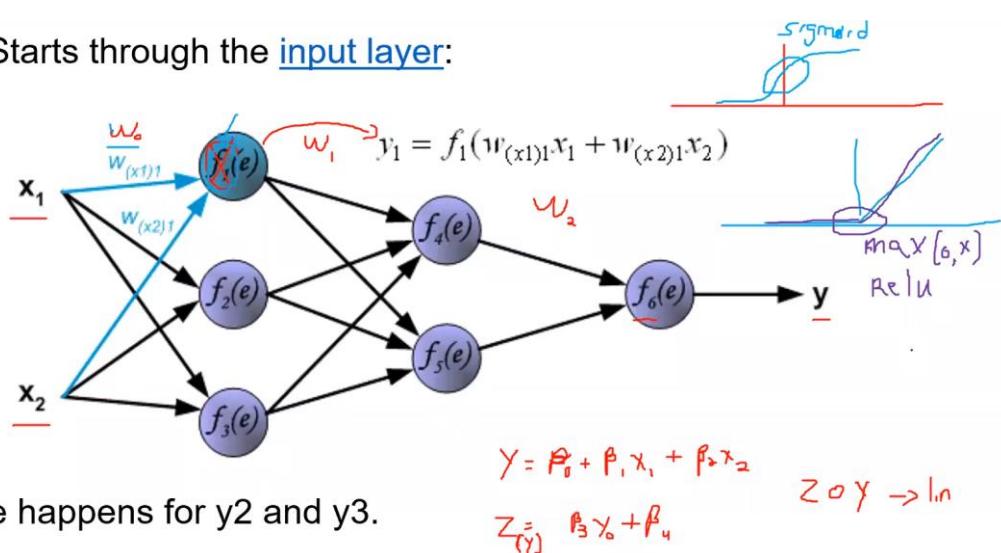


- With some assumptions about the activations between layers
(Activations must be non-linear as a composition of linear functions is also a linear function)

רשת בסיסית:

air נראית רשת בסיסית? נראה הממחשה קטנה.

- Training Starts through the input layer:



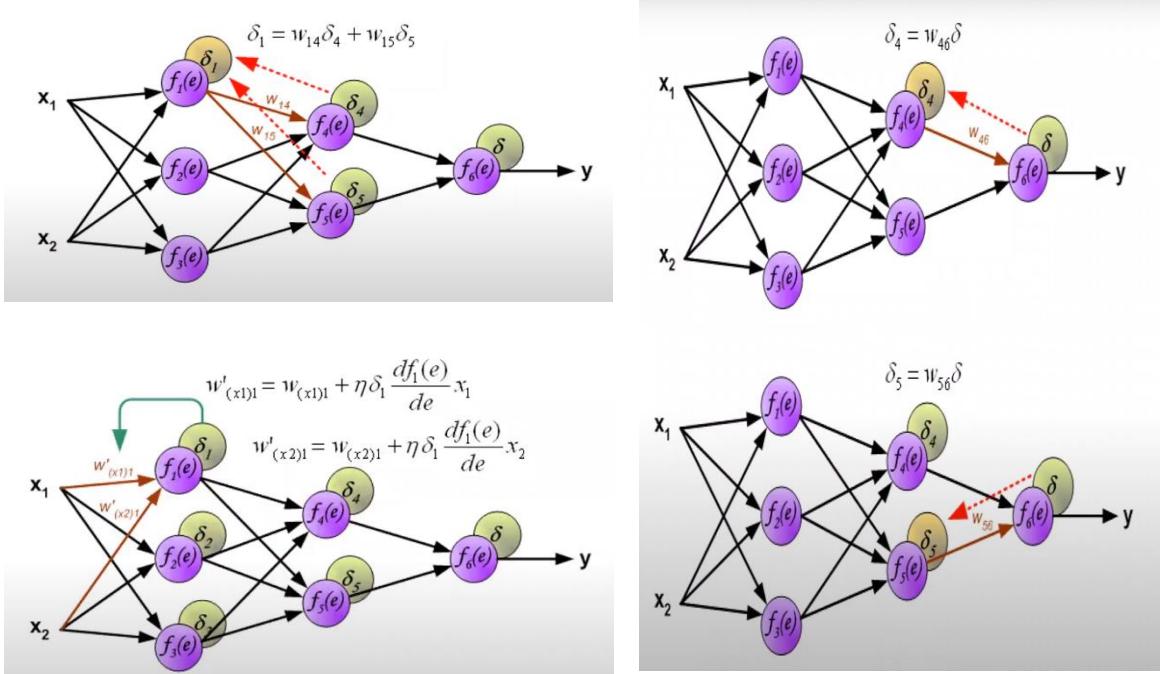
- The same happens for y_2 and y_3 .

על הנירון אנחנו מפעילים פונקציית אקטיבציה (f) שמטרתה להכניס או לנארוות לתוך התהליך, הסיבה לכך שאם אנחנו רוצים לייצר רמת מרוכבות גבוהה יותר, אנו נרצה ליצור אלמנט שהוא לא לינארי.

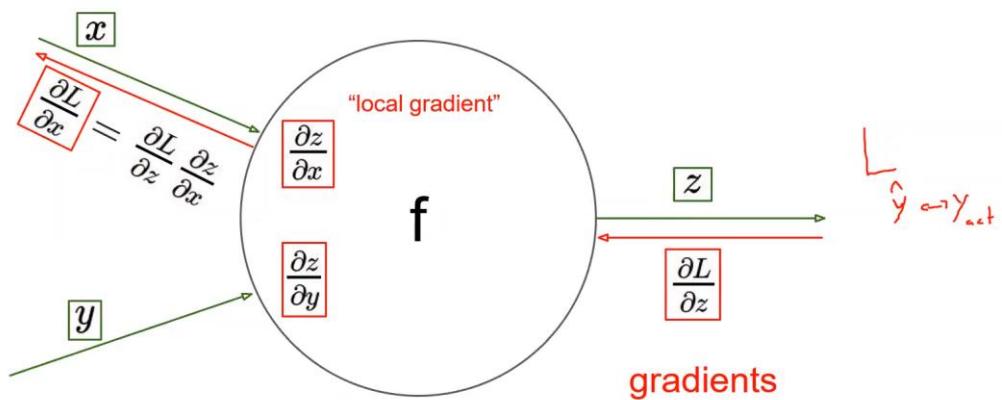
כעת לאחר קבלת הפלט אנחנו נרצה ללמידה ולשפר את התוצאות, בעזרתו פלט אנחנו נרצה לעדכן את המשקלות שלנו על מנת לגרום להתקאה של המודל.

בעצם מחשבים פונקציית הפסד במקורה שלנו הפלט מול הפלט המצופה. וכעת נחשב את הנגזרת של פונקציית הפסד הזאת ביחס לכל אחת למשקלות שהיא לנו בתחום. בשלב הראשון בעדכן של המשקלות "האחרונות" של הרשת היא ד"י פשוטה אבל ככל שנלך לתחלת הרשת אנחנו צריכים לעדכן את המשקלות על פי המשקלות שלפניין (בסדר הפור כמובן – מהסוף להתחלה) אז משתמש באותו chain rule כדי לקשר בין הנגזרות ההתחלתיות לנגזרות החלקיות שיש לנו ביחס לכל אחת מן המשקלות.

נרצה בעצם לפעוף את אותו הפסד שקיבלנו בסוף הרשת ביחס לכל אחת מהמשקלות.

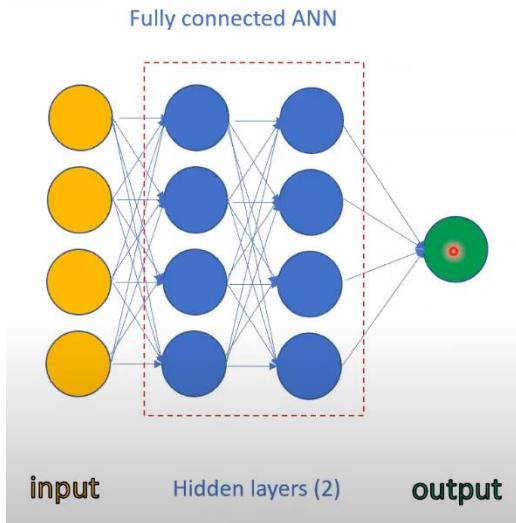


על מנת להסתכל על הגרדיינט של Z ביחס לא אנחנו נסתכל על הנגזרת החלקית של Z ביחס לאו ואותו דבר גם לי ומה שנקבל זה הנגזרת של הפסד ביחס של פונקציית ההפסד L שאנו מחשבים אותה ביחס לבין התוצר שלנו לבין התוצר שלו אנו שואפים להגיאו.



:The building blocks

Layers -



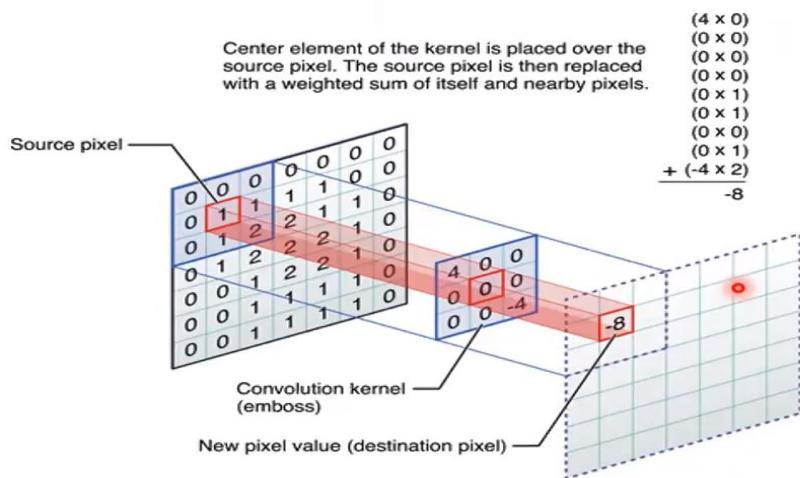
כרגע אנו מדברים על רשתות ספציפיות שנקראות **fully connected networks** שבהן כל ניירון מחובר לכל השכבות בשכבה העוקבת. בשכבה האחורונה אנחנו נקרא שכבת הפלט (כמונן שיכול להיות מספר פלטים ולא ניירון פלט בודד).

סוג נוסף של רשתות שמתעסק בהן הן רשתות קונבולוציה (CNN), קונבולוציה הינה פעולה שמייצרת או נובעת ממתן חשיבות גבוהה יותר לחבר בין אלמנטים קרובים בקלט, מה זה אומר? אם עד עכשיו שדיברנו על **fully connected network** אם היו לנו מספר ניירונים שמציגים את הקלט (לדוגמא תמונה בגודל 7×7 含有 49 פיקסלים אז כל ניירון היה מייצג את אוטם הפיקסלים ומחבר לכל שאר המניירונים בשכבה החוביה הבאה).

הרעilon ברשתות CNN שאנו נאנו לא לחברים הכל להכל אלא אנו רצים למוד פילטרים (מטריצת משקولات), כשהאנחנו נפעיל את הפרמטרים האלה כדוט פרודקט (כלומר כפל של איבר איבר מתוך הקלט, כמו בשקוף שאנו רואים) אז התוצאה תהיה לנו ייצוג חדש של הנתונים.

את אותה פעולה נбурיר על כלל הקלט שלנו ובסופה של דבר נקבל ייצוג חדש של הקלט שיאפשר לנו לייצג טוב יותר את הקלט ביחס לבעה נתונה. מטרת הייצוג החדש אינה בא בשbill לדוחס את הקלט או לייצג אותו באופן יותר עיל, אלא מיפוי בין הקלט המקורי לאיזשהו פלט רצוי.

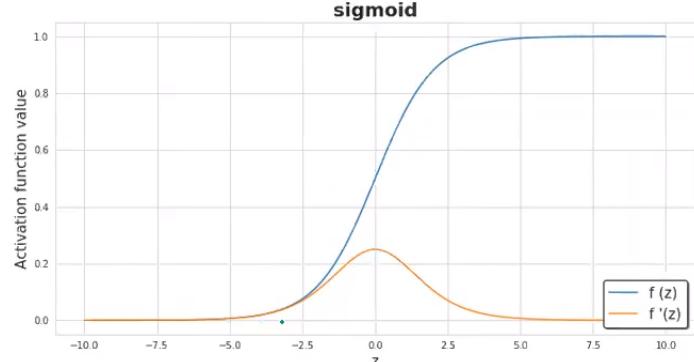
Convolutional Neural Network



Activations -

פונקציות אקטיבציה שונות, ראיינו שאט **pid Sigmoid** ניתן בקלהות לגזר אותו בשבייל לפעוף אותו לשכבות הקדומות של הרשת.
בדומה ל^{pid} Sigmoid יש לנו פונקציה דומה בשם **tanh** וניתן להגיד ששנייה שייכת למשפחה קרובה.

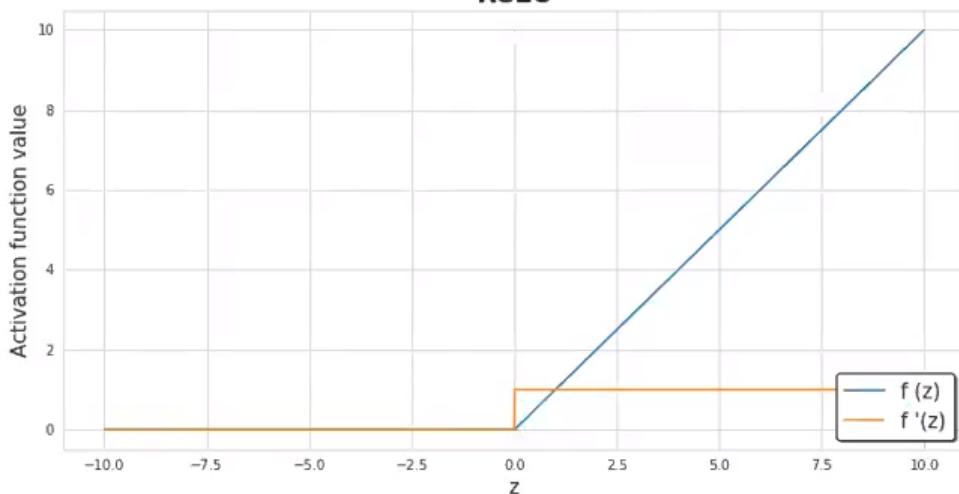
אחד הביעות שראינו בפונקציית **Sigmoid** היא שהגדיאנט של הפונקציה הוא מעורבל מאוד.



הוא אף פעם לא שלילי ובמקסימום מגע לאיזה 0.24 דבר שייגרום בעיה בעיקר כשנרצה לעשות הרכבה של הרבה פונקציות כאלה.
כל שיש לנו יותר שכבות שעליין נפעיל פונקציית **pid Sigmoid** בתור אקטיבציה אנחנו נקבל גרדיאנט שהולך ונעלם, ככלmore בשכבות הראשונות בשלב הפעוף אנחנו כן נעדכן את המשקלים אבל בשכבות המתקדמות המשקל כמעט ולא ישנה.

דבר נוסף שחייב של **Sigmoid** באופן חזר המון פעמים יכול להיות חישוב מאד מסובך, לנו נעדיף להשתמש בהרבה מקרים בפונקציית אקטיבציה מסווג **ReLU** שהגדיאנט שלה הוא 1 כאשר מדובר במספר חיובי, ו 0 כאשר מדובר במספר שלילי.

ReLU

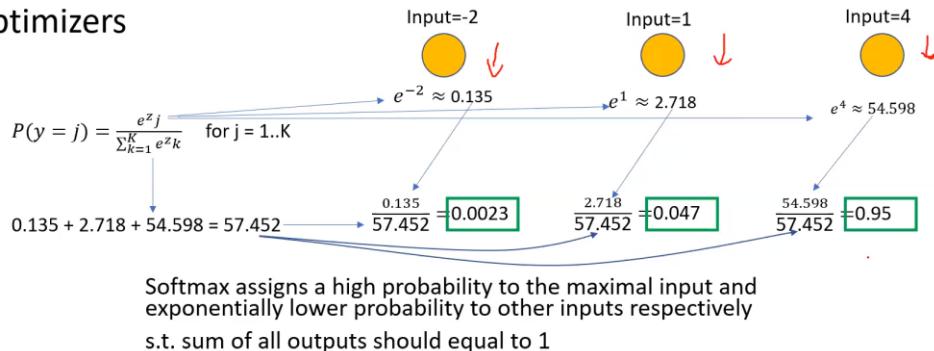


פונקציית אקטיבציה נוספת שנטען בה הינה Softmax בשונה ממה שראינו עד עכשוו, פונקציה זאת מייצרת לנו אפשרות הסתברות שאינה ברמת נוירון/קלט בודד אלא ברמת כל הקלטים באוטה שכבה.
מה זה אומר?
נמיר קלטים שונים להסתברות להיות במצב מסוים.

דוג' :

נגיד יש לנו שלושה קלטים -2, 1 ו 4 ואנחנו רצים לתרגם את הקלטים האלה להסתברות להיות בכל אחת מן המינים הנ"ל, אז נרצה לייצר משקל שמתאר איזהו סיכוי שמתאר שאנחנו נמצאים במצב שהקלט הוא גבוה יותר (4), ובכל זאת לייצר משקל גם למספר שליליים או מספרים מאד קטנים.

- Layers
- Activations
- losses
- Optimizers
- Softmax (cont.)
- Number of nodes is equal to number of categories
- Lets explore an example for 3 categories:



פונקציית Softmax טוביה כשאנו רצים לייצר מצב של הסתברויות, בשביל לייצר מצב של אני נמצא או פה או פה, ולא מצב שאינו יכול להיות גם וגם. מתאים לביעות של מולטי-קליאס קלאסיפיקשיין.
כמו כן, בדרך כלל משתמש בשכבה האחורונה בשביל לקבל הסתברות לגבי המצב שאנו נמצא בו.

Loss function

פונקציות הפסד שנוהגות להשתמש בהן לביעות ריגרסיה:

- Mean-squared-error (MSE) – נשתמש כשןרצה להקטין חיריגות מאוד גדולות.
- Mean-absolute-error (MAE) – נותן ערך יותר גדול לסך הטעויות אבל פחות לטעות קיצונית באופן יחסית לשאר הטעויות.

$$\text{MSE} = \sqrt{\frac{1}{n} \sum (y_i - \hat{y}_i)^2}$$

- Mean-squared-logarithmic-error (MSLE)
- Mean-absolute-percentage-error (MAPE)

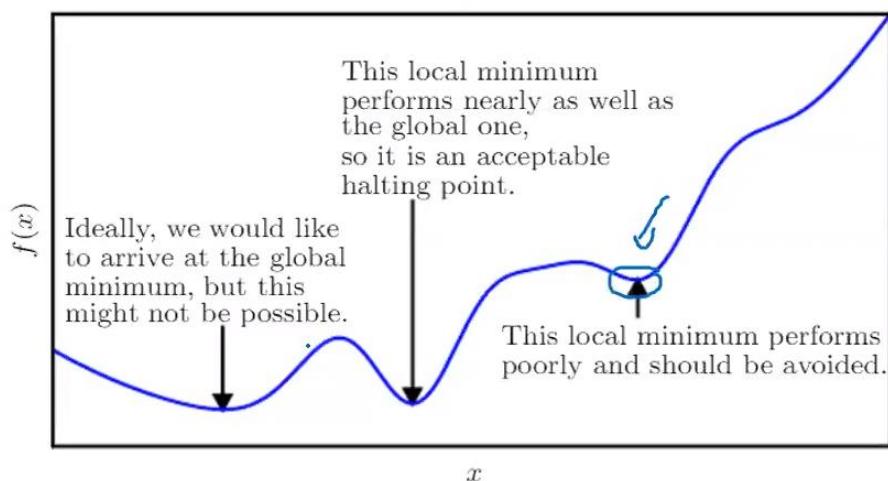
$$MAE = \frac{1}{n} \sum |y_i - \hat{y}_i|$$

פונקציות הפסד שנהוגות להשתמש בהן לביעות קלאסיפיקציה:

- Logistic loss (binary cross-entropy)
- (יתנה די דומה למה שראינו בsoftmax) Categorical cross-entropy

Optimizers

הרעין הכללי אומר שנוכל להשתמש באופטימיזר בשביל להגעה לפתרון האופטימלי, הבעה היא שיש לפעמים מינימום מקומי שאנחנו עלולים להיתקע בו, במצב צזה אנחנו נרצה לצאת ממנו ולהצליח להגעה למינימום הגלובלי או לפחות למינימום אחר מאוד קרוב לאותו מינימום גלובלי אופטימלי.



Gradient Decent ○ – אנחנו מחשבים את הגרדיאנט בהתחשב בכלל הדוגמאות בנתונים שלנו, אנו מחשבים את הפסד ומתקדמים בכיוון הגרדיאנט המוצע ביחס לכלל הנקודות.



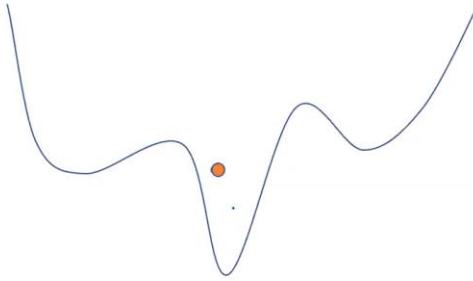
אם eta קטן מדי אז זה פשוט יקח יותר מדי זמן.

אם eta גדול מדי זה יכול ליצור מצב של התבדרות ולא נגיע להתקנסות. לכן נשאף למצוא את eta הנכון על מנת להגעה להתקנסות נכונה.

שימוש במומנטום , אגדציה של הארגדיינטים הקיימים.

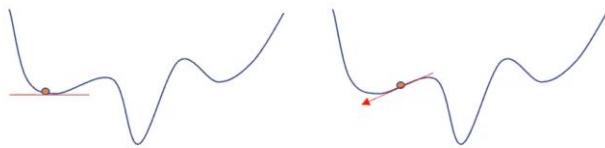
One solution to this problem is using **momentum**

We can think of momentum as the rolling ball acceleration – mathematically we add the decaying average of previous gradients to our current gradient



This allows us to avoid two causes of local minimum convergence:

- Plateau with zero gradients and no more updates
- Oscillations caused by opposite gradients

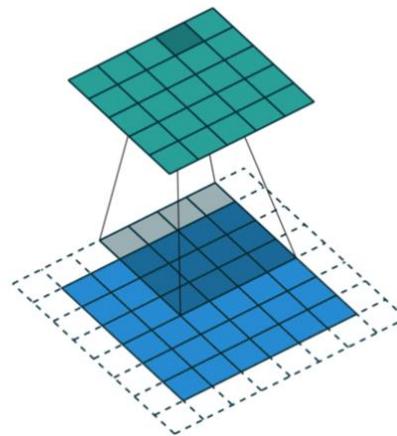


רוב הזמן אנחנו נעבד עם אופטימיזר מסווג ADAM

- **RMSProp** - adaptive learning rate per weight, learning rate is divided by $(\sum \exp \text{decaying prev. grad}^2)^{\frac{1}{2}}/n$
- **adam** – RMSProp + momentum + bias correction

הרצאה 3

חזקה קצירה על רשות CNN, בעצם אנחנו מבצעים תהליך שבעזרת הפילטר אנחנו מוציאים את המידע באופן שונה.



וכעת דוגמה של כמה פילטרים שמתאימים לצורה שונות (מאונך, אופקי)

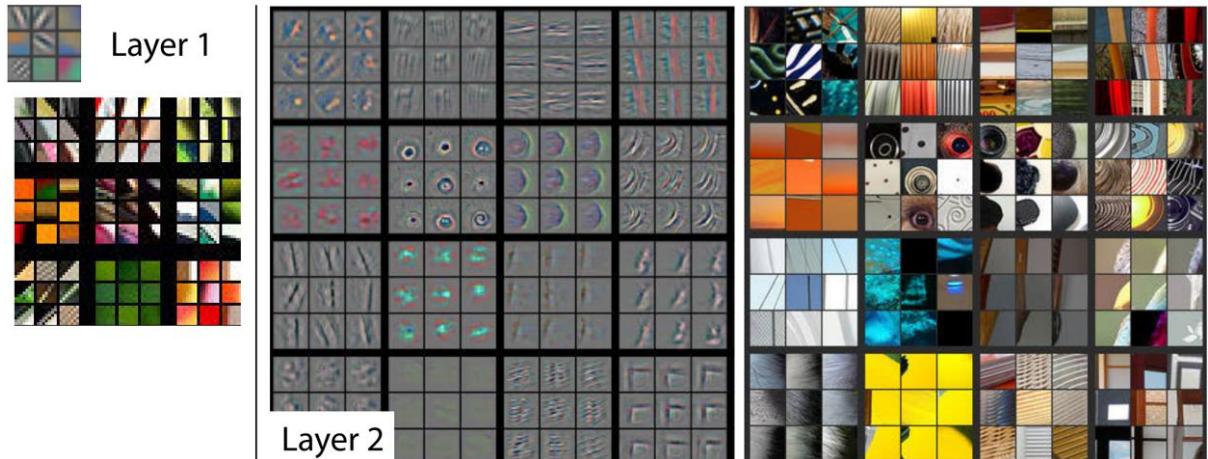
*פילטר אופקי

ה'תל

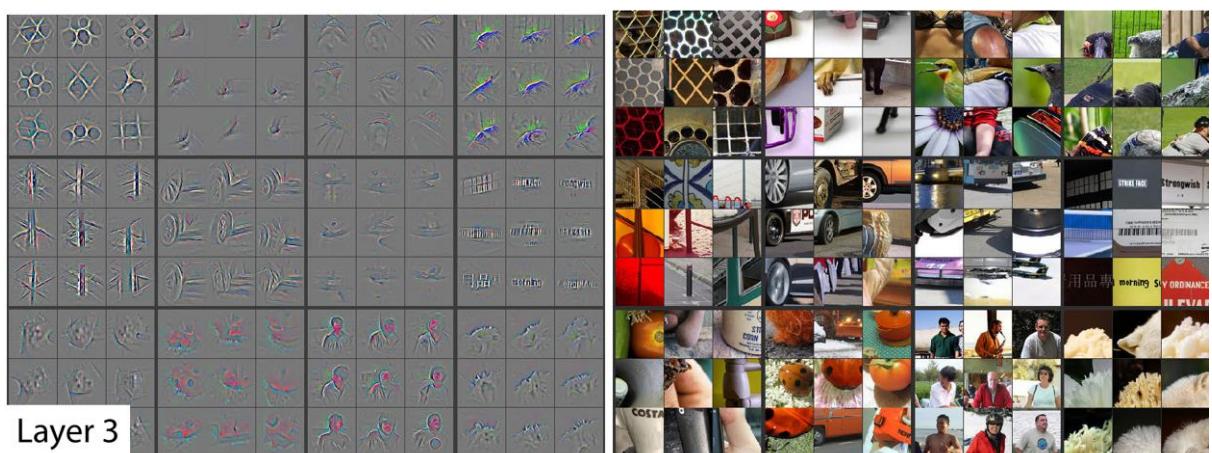
うか

Understanding convolutions

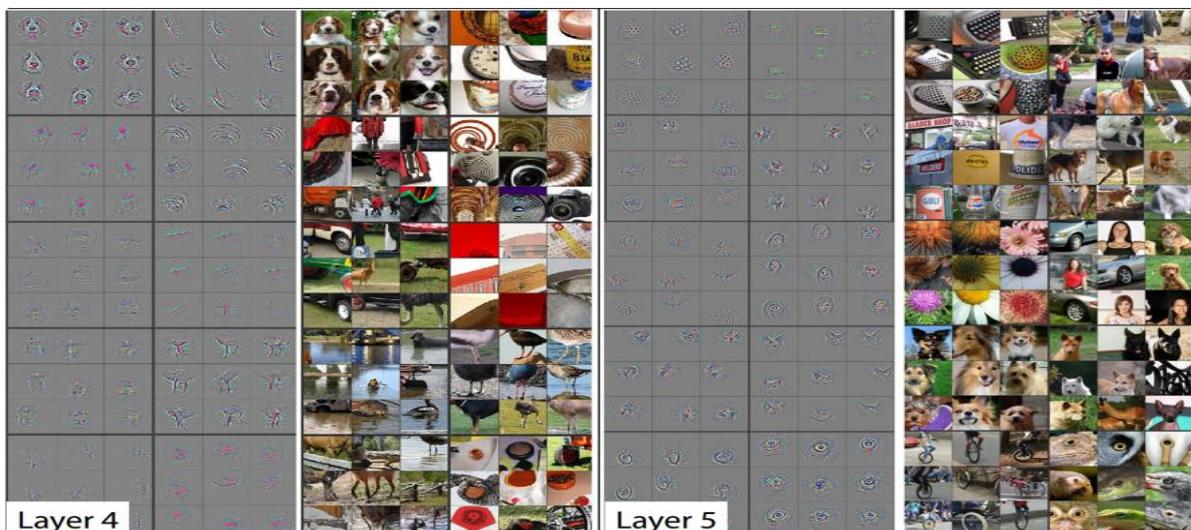
דבר ראשון נרצה להבין כיצד רשתות מתקדמות אז נתבונן בתמונה הבאה:



אנו רואים איך בשכבה השנייה אנחנו כבר מקבלים לתמונות באופן לא רע, אם אנחנו מצלחים כבר לזהות צורות כמו מעגלים או קוויים מעוקלים וכו'



כאן כבר ניתן לראות בצורה יותר מודנת צורות ברורות ואפילו כבר>Zיהוי של פרצופים זהה נתון לנו אינטואיציה די' טובה שאפשר כבר ממש לזהות אובייקטים מורכבים על גבי התמונות.



ובתמונה الأخيرة אנחנו כבר רואיםشبשכבה הרכעית והחמיישית יש אפילו זיהוי של פנים חיות שונות, והפילטרים רק הולכים ומשתפרים בין שכבה לשכבה ונهاימ יותר מרכיבים.

VGG16 architecture

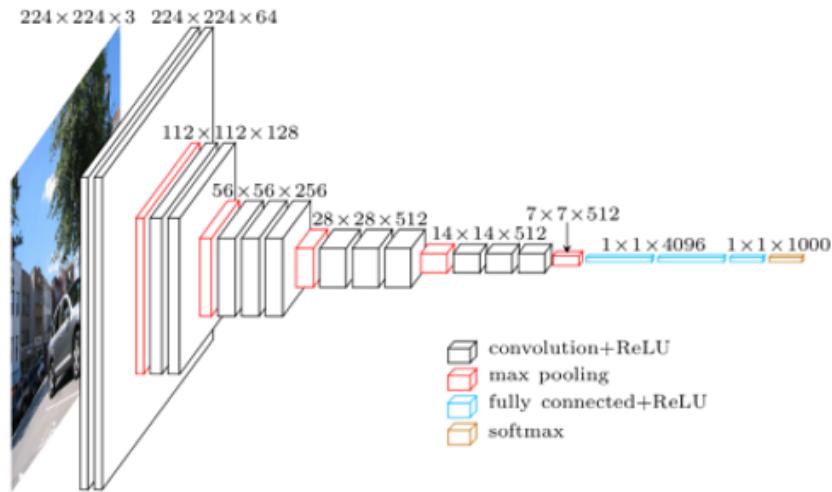
אריכטטורה מאוד פשוטה אבל עם זאת הביאה לתוצאות מאוד טובות בשנת 2014, בניית בבלוקים באופן ש חוזר על עצמו.

- שתי השכבות הראשונות בעלות שתי שכבות קובולציה ואז מוצעים מקס פולינג
- שלושת השכבות לאחר מכן בנויות משלוש שכבות קובולציה ואז מוצעים שוב מקס פולינג
- והשכבה الأخيرة לוקחת את הפלט מהמקס פולינג מהשכבה לפנוי, משפח אותו ומוסיף מעליו שתי שכבות fully connected עם 4096 נוירונים, ולאחר מכן שכבת Dense נוספת עם 1000 נוירונים. (1000 נוירונים כי במקרה הספציפי זהה היו 1000 מחלקות שונות).

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312
predictions (Dense)	(None, 1000)	4097000

Total params: 138,357,544
 Trainable params: 138,357,544
 Non-trainable params: 0

הלו



נשים לב שבשכבה האחורונה יש לנו מעל 120 מיליון פרמטרים וזה יותר מכל שאר השכבות האחרות ביחד. זה איזהו אלמנט שיוצר מודל מאוד כבד עם אימון ארוך מאוד, בהמשך אנחנו נראה פיתוחים נוספים ושיפורים כיצד למנוע את הדבר זהה.

הסבר קצר לפיצ'ר אקסטרקטור:

נניח אני רוצה לבודא ולזהות בני אדם לפי מוצא אתני, וכרגע אין לי מודל שיודיע לעשות את זה, מה שאני כן יכול לעשות זה ללקחת מודל שכבר אומן וידע נגיד לזהות צורות ואיפילו יודע לזהות בני אדם בתמונות, יוכל להשתמש במודל הזה כנקודת התחלה במקום לבנות או ליצור מודל חדש מאפס שאני אצטרך לאמן אותו מחדש על מנת שיבין כיצד לזהות בני אדם. (pretrained model.)

:Transfer Learning

3 יתרונות עיקריים:

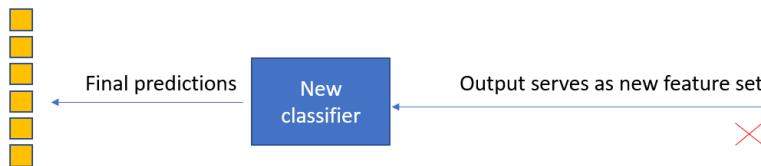
- להגיע ל`convergence` בצורה מהירה יותר
- להקליל בצורה טובה יותר (`improved generalization`)
- להגיע לתוצאות יותר טובות מאשר לנו מעת מוד דוגמאות מתויגות.

לא תמיד נוכל לעשות זאת בצורה חלקה, לא תמיד יהיה לנו רשות שאומנה על משזה מספיק קרוב אז דומה למה אני ציריך, לדוגמה אין לי רשות שאומנה על וידאו, אבל אם יש לי רשות שאומנה על תמונות אני יכול להסתכל על הוידאו כפריים בודדים ואז להשתמש בהאותה רשות שאומנה על תמונות.

על מנת להשתמש בפיזר אקסטרקטור נוכל ללקח לארכיטקטורה הקיימת שלנו רקחת איזה פלט פנימי שלה סוג של שכבת בינאים (לא דווקא הפלט הסופי של אותה הרשות)

דוגמא בתמונה:

- Use the last layer (or any other informative layer) as output instead of the original output
- Train a new classifier on these extracted features (the new graph's predictions)



Layer (type)	Output Shape	Param #
Input_2 (Inputlayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 112, 112, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
Flatten (Flatten)	(None, 25088)	0
Tc1 (Dense)	(None, 4096)	102764544
Tc2 (Dense)	(None, 4096)	16781312
predictions (Dense)	(None, 1000)	4097000

Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0

נוותר על השכבות הפרדיקציות (האחרונה) ושכבה מעל נוציה את הפלט על מנת להשתמש בו כפיזר קללאסיפיר חדש.

AlexNet architecture

הארquitektורה של AlexNet יסית פשוטה, (הוא קצר מפורט על זה אבל גם הדוגמה שיש במצגת היא לא הרשת המקורית, אלא רשת עם קצר הקלות מתוך התחרבות של חוסר GPU)

יש לנו **input** בגודל של 3, **224X224X3** מסמל את RGB . על הקלט זהה אנחנו מפעילים **קרנל אבולוציוני** בגודל **11X11**.

– stride מגדר את מרוחק הקייצה, כאן מגדר ב-4, כלומר כל פעם קופצים בהזחה של 4 פיקסלים. היות וגודל הקרן הוא גדול, אנחנו לא רצימ ליציר הרבה overlap וזה מאפשר צמצום של כמות החישובים.

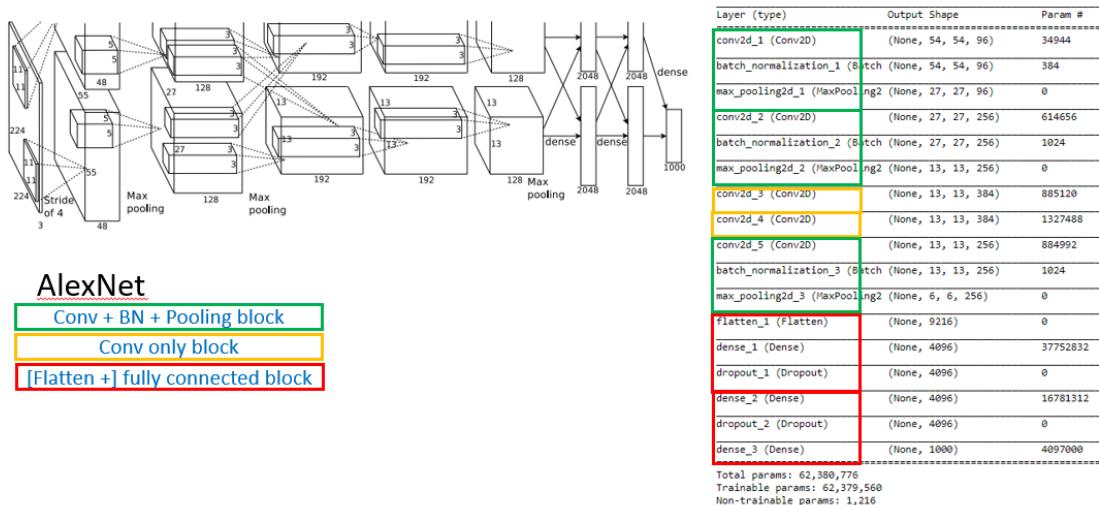
אנחנו רואים **שהרשת מפוצלת**, הפעולות באפיק העליון והתחתון זהות, אך הארכיט' מפוצלת לשניים במנורה לרץ על 2 GPUs בנפרד, כי משאבי החישוב היו מוגבלים.

יש לנו 48 filters בכל אחד מהערוצים זה אותו split שדיברנו עליו. כאמור, הם מפוצלים. בлок ירך חוזר על עצמו שלוש פעמים, בסוף הארכ' אנחנו משחחים את הקלט באמצעות flatten , כך שנקל וקטור בגודל 9216. הקביעה זו יסית אקראית ולא נובעת מסיבת ברורה.

בצד ימין מופיעות כמה פרמטרים שככל אחת מהשכבות מחזיקה. ניתן לראות שכבת DENSE הראשונה בעלת **37 מיליון פרמטרים** כי אנחנו למשה מושחים את הקלט ומחברים הכל להכל, אורקלוחב עמוק X כמות הנוירונים בשכבה fully connected, הכולמר אותם 4096 נוירונים בשכבה יוצרת כמה אDIRה של פרמטרים.

בשכבה הראשונה של fully connected יש כמעט 37 מיליון פרמטרים. בשכבה הבאה, יש כמעט 17 מיליון פרמטרים וסה"כ בראשת יש לנו מעל **62 מיליון פרמטרים**.

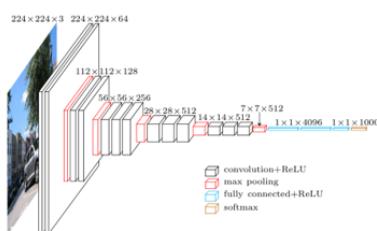
נזכיר **Drop out**, מושמיים חלק מהקלט באופן רנדומלי, בעצם מסתירים חלק מהקלט שעבר מהשכבה הקודמת אבל בצורה אקראית. ככלمر ניתן לחשב על זה כמה מסתירים חלק מהsamples הרעיון הוא שאנו רוצים להימנע מ מצב שהרשת תלמד שם פיקסל מסוים הוא במיקום מסוים, אז אנחנו רוצים שהיא תעתלם מפיקסל במיקום מסוים ולכן אנחנו רוצים שהרשת תלמד מסלולים שונים, מזכיר – Bagging אנחנו מוצעים עץ החלטה בכיוון שונה, הרעיון כאן דומה – אנחנו נעים בכיוון הממושע, אנחנו כל פעם ממסכים את האובייקט בצורה שונה, כך שבאופן כללי נלך במסלול הנכון. זו הייתה יריית הפתיחה לIMAGE!.



VGG19

הבדל העיקרי בין VGG16 ל-VGG19 הוא בבלוקים המסומנים בצבע צהוב שב-VGG19 יש לנו 3 פעולות קונבולוציה ולאחר מכן פעולה של MaxPooling לעומת 4 פעולות Konvoluzija ורף לאחר מכן פעולה של MaxPooling.

אנחנו יכולים לראות שגם מבחן הפרמטרים אין הבדל מאד גדול, וניתן להגיד שגם ברמת הדיווק ההבדל לא כל כך גדול, הוא כן קיים אבל אינו כל כך גדול.



VGG16

- 2 x conv 1 pool block
- 3 x conv 1 pool block
- Flatten + fully connected block

VGG19

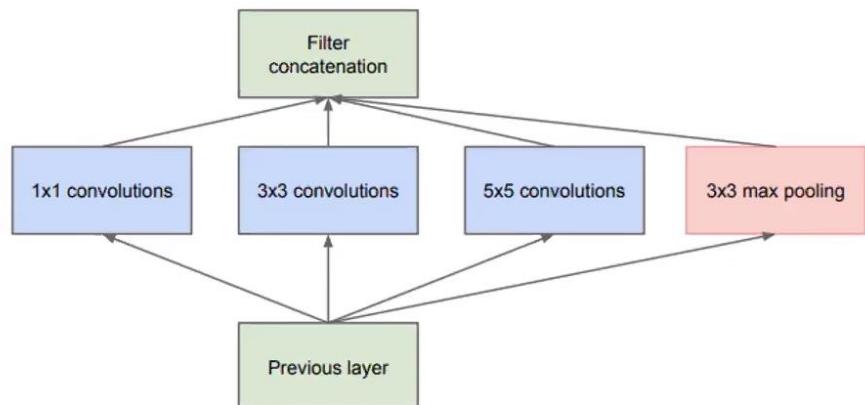
- 2 x conv 1 pool block
- 4 x conv 1 pool block
- Flatten + fully connected block

Layer (type)	Output Shape	Param #	Layer (type)	Output Shape	Param #
Input_2 (Inputlayer)	(None, 224, 224, 3)	0	Input_4 (Inputlayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792	block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928	block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0	block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856	block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584	block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0	block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168	block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080	block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080	block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0	block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160	block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808	block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808	block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0	block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808	block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808	block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808	block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0	block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0	flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544	fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312	fc2 (Dense)	(None, 4096)	16781312
predictions (Dense)	(None, 1000)	4097000	predictions (Dense)	(None, 1000)	4097000
Total params: 138,357,544					
Trainable params: 138,357,544					
Non-trainable params: 0					
Total params: 143,667,240					
Trainable params: 143,667,240					
Non-trainable params: 0					

Inception/ GoogLeNet

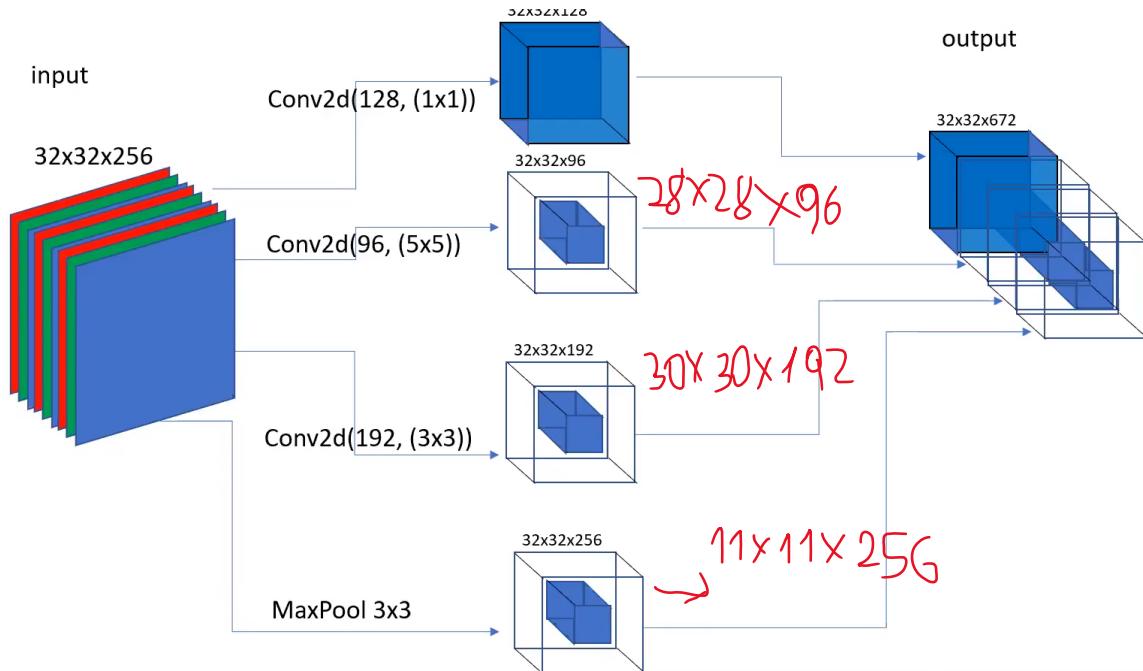
הרעין הבסיסי בארQUITקטורה היזאת היא בלוקים בסיסיים בדומה ל-VGG שוחזר על עצמו, רק שאוותם בלוקים בסיסיים יותר מורכב.

למה להסתכל רק על קונבולוציות מסדר גודל של קernal 3X3 שמה שאנו יכולים זה להסתכל גם על קונבולוציות בגודל 1X1, 5X5 ו-3X3 ואנו כבר אז גם להסתכל על פעולה של MaxPooling בגודל 3X3.



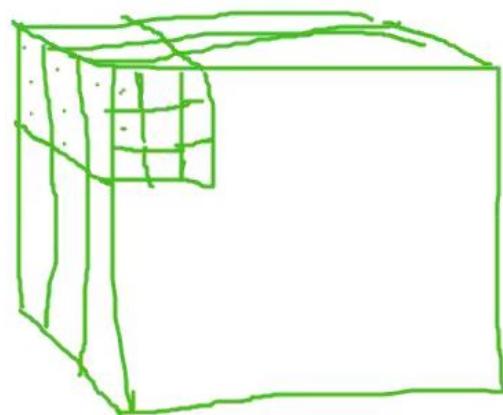
(a) Inception module, naïve version

בעם בכל בלוק אנו מקבלים קלט שהוא תוצר של הפלט מהשכבה הקודמת, ועל הפלט זהה אנחנו מבצעים במקביל כמה פעולות שונות (במקרה זה 4), ואת הפעולות הללו אנחנו משרשים מעבר לפלט.

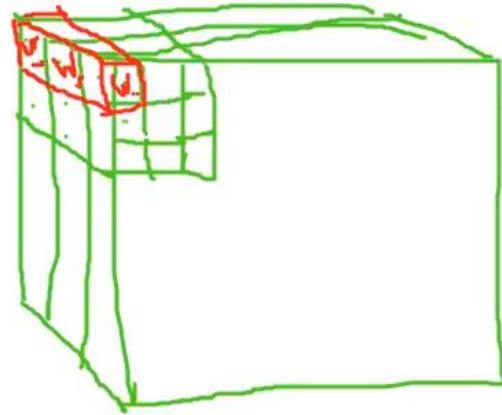


למה שבעצם נעשה קונבלוציה בגודל של 1×1 ? הרי אם מה שאינו לומד זה לא קשור בין פיקסלים סמוכים אז מה המשמעות של אותה קונבלוציה בגודל זה?

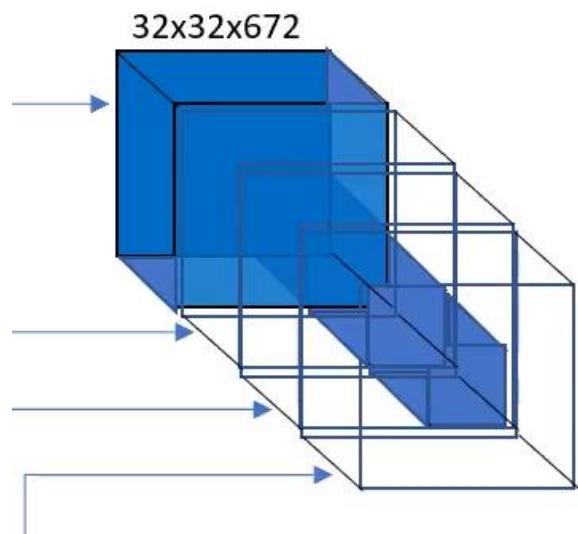
שראינו פילטרים בגודלים שונים כמו 3×3 אז בעצם מה שהיינו עושים זה להעביר את הפילטר בשבייל ליצור קשרים בין פיקסלים סמוכים, כמובן שגם גם מדובר על מרחק העומק, אז אם הפילטר בגודל $3 \times 3 \times 256$, זה דוט פורדקט של כל נקודה בפילטר שלנו וכל זה ועוד כל שאר הנקודות לאורך השכבות שלנו.



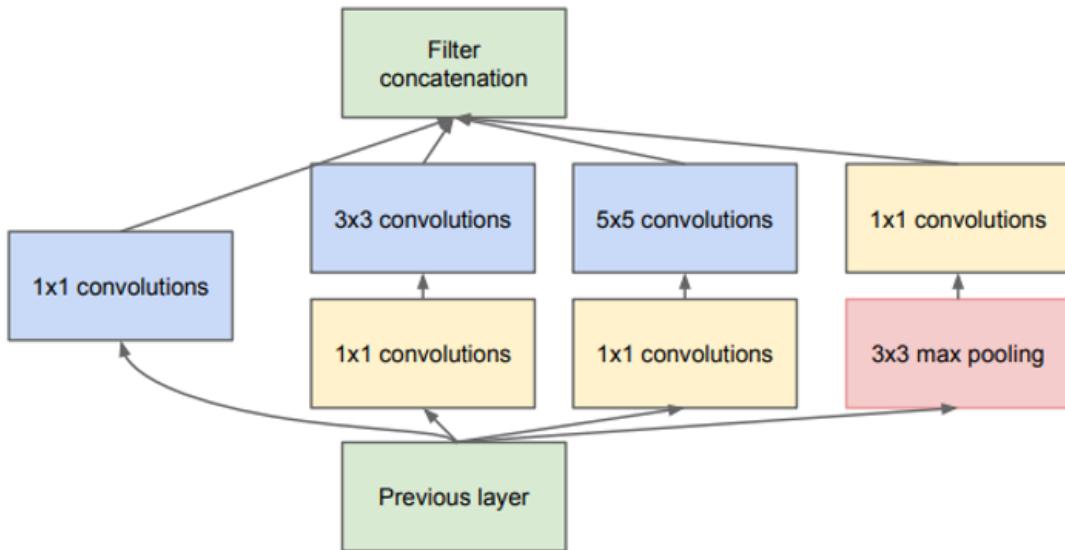
עכשו בReLU של 1×1 אנחנו בעצם לא מסתכלים על פיסקלים סמוכים אבל הוא כן ממילא לנו את הפלט הפיצ'ר מאפ השוניים. בעצם נלמד את המשקל היחסי של הקלט שלנו בהיבט של הפיצ'ר מאפ השוניים שלנו.



בחזרה לבlokים שלנו, כמוון שעל מנת לשרש את כל התוצאות ייחדי נצטרך להתאים את הגודלים שלהם, ולכן אוטם באפסים בשבייל יהיו יהיו בגודל אחד, כל אחד בהתאם בכמה שחסר לו בשבייל לחזור להיות בגודל (עומק-הרזי-לא-רלוונטי) $32 \times 32 \times 32$.



במחקר על אותה רשת, החוקרים גילו שיש איבוד רב של משקלות בשיטה הנאיית, لكن הם רצו לצמצם את האיבוד של אותן משקלות, והדרך שלהם לעשות זאת הייתה באמצעות קונבלוציות בגודל של 1×1 באופן הבא:



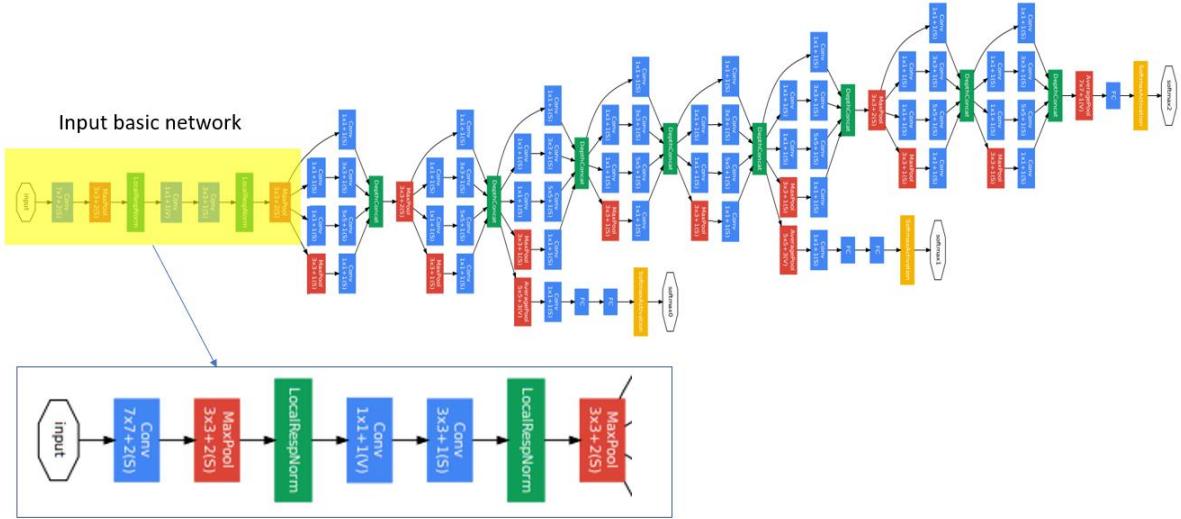
(b) Inception module with dimension reductions

از אם ראיינו בדוגמה הקודמת שהשתמשנו בפילטר של 1×1 על מנת למשקל בין הפיצ'ר מופיע השונים, כתה נראה שיש שימוש נוסף בפילטרים מסדר גבוה של 1×1 שהוא הורדת ממדים, זאת אומרת שאם קיבלנו קלט שהוא מאד עמוק מבוצן של ערכאים, אנחנו יכולים לצמצם את הממדיות שלו באמצעות כמה מוגבלת של פילטרים בגודל של 1×1 .

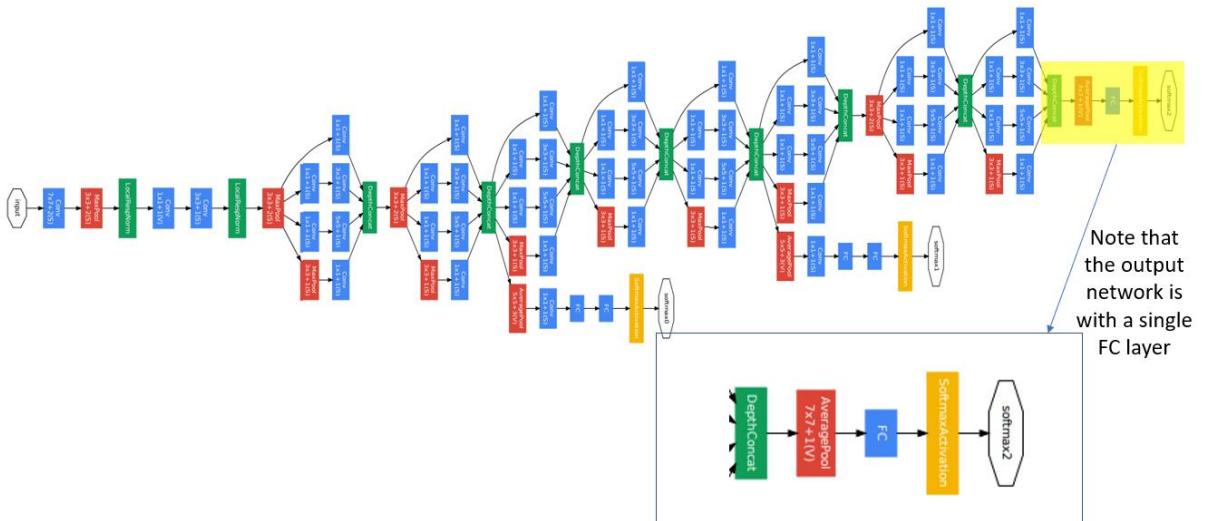
בגדול אנחנו יוצרים קומבינציה לינארית של פיצ'רים שקיבלו מהשכבה הקודמת, אנחנו נלמד את שכבות הקונפלוציה על אותה קומבינציה לינארית שגם היא עצמה נלמדת.

הסיבה שעושים פילטר 1×1 אחרי הפעולה של ה-*Maxpooling* זה שאין סיבה ללמידה משקל על מידע שגם ככה לא ישרוד, אז נשתמש בפילטר רק על המידע שנשמר/שרד את הפעולה הזאת.

ראשית Input basic network
 אנחנו מבצעים קון' בגודל 7×7 , ואז MaxPooling ולאחר מכן גרמול של הוטקנו.
 לאחר מכן, שתי פעולות קונבולוציה ואז גרמול ובסוף שוב MaxPooling.



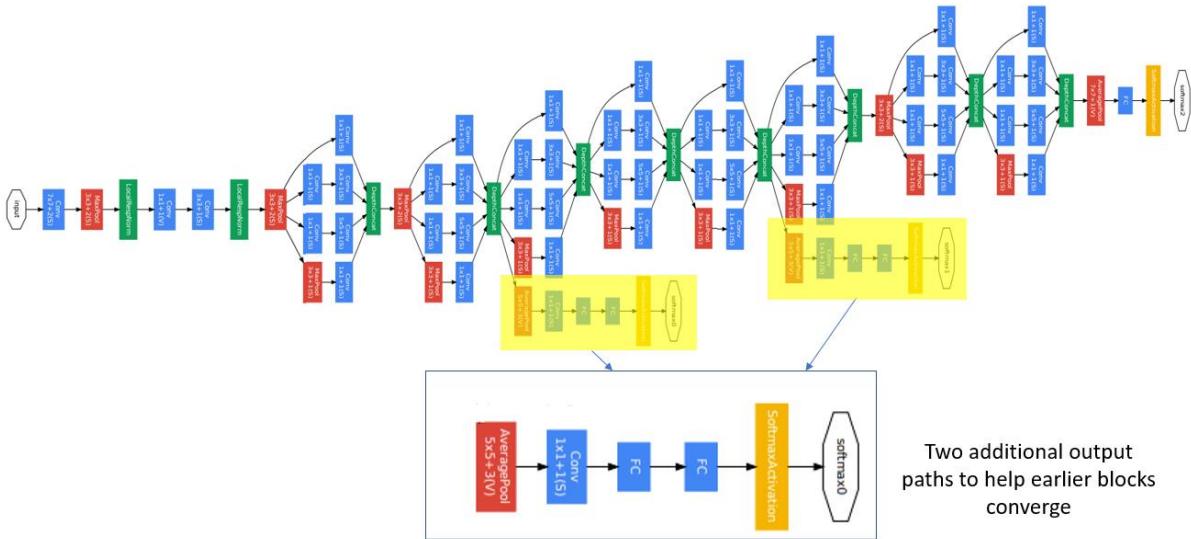
בסוף יש לנו אלמנט חדש שלא דיברנו עליו, שנקרא **global avg pooling** - עברור כל feature map ישר לקיבלי אני מהציג מספר בודד.
 כאן אנחנו לא משתמשים את **feature map** אנחנו ממצאים אותו.
שאלה: מה היתרונות של פעולה זו? מצומצם של הממדים באופן משמעותי, לפני ה - **fully connected**, וכך אנחנו מקבלים הרבה פחות פרמטרים בשכבה הבאה.



חסרונות בשימוש בפעולת global avg pooling?
 שכן אלמנט חדש שמאפשר מצומצם משמעותי של הקלט לשכבה ה - **fully connected** אבל אנחנו עלולים לאלץ הרבה מידע בפעולת זו.

עוד בעיה נוספת – בעיות עומק – קושי בפעוף הגראדיאנטים לשכבות הקודמות.
 החישוב של ה output מתרחש בשכבות עמוקות של הרשת ואני נתקלים בעיה בפעוף אחריה של הגראדיאנטים וזה אונסיפים בשלב מוקדם יותר את ה output של הרשת, לאחר מכן אנחנו מחשבים את ה Loss וכן זה תהליך קצר יותר. כמובן ובוצע את החישוב של LOSS מאוחר

למידה פחות עמוק, זו למשה אינדיקציה חלקית.
בעםם הם הוסיף עוד שני בלוקים של Output וממה שיאפשר לשכבות המוקדמות להתכנס.



Inception v3

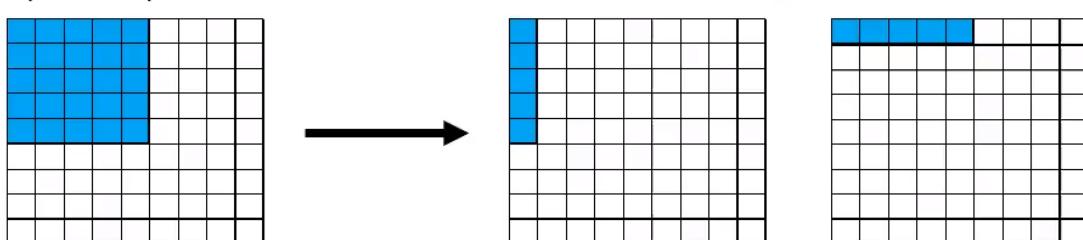
ב-3v אנחנו נחשפים לרעיון שבמוקום שנשתמש בקונבלוציות מרובעות, משתמש בקונבלוציות שנקיימות flatten convolutions כלומר בקונבלוציות מסדר 1 ax, ax. 1.

הרעיון שברגע שנשתמש בקונבלוציה מסוג ax 1 מיד לאחריה נשתמש בקונבלוציה בגודל 1 ax, לדוג' עבור קונבלוציה מסדר 5X5 אנחנו מאמנים 25 משקלות, כשאנו עונים לkonvolוציה מסדר 1 5X1 ועוד 5X1 אנחנו מתייחסים לאותו אזור, מצד שני אנחנו פחות מחצ'י מהמשקלות.

זה לא היה נכון להגיד שאנו מייצגים את אותם דפואים של מידע, אך אפשר להגיע לרמת קירוב מאוד קרובה של אותם דפואים.

Instead of having 5x5 computations per channel

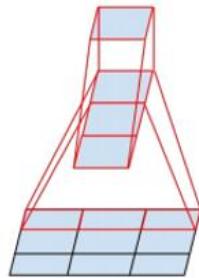
We can have 5+5 computations per channel and still represent most of the signal in the data



Vertical flattened convolution followed by horizontal flattened convolution

דבר זה יכול לעזור בשלב ההתקנות אבל יש פה off trade off, ההתקנות תהיה מהירה יותר אבל אנחנו בכל זאת מ Abedim במידה מסוימת את יכולת הייצוג של המודול.

תיאור של המבנה:



type	patch size/stride or remarks	input size
conv	$3 \times 3 / 2$	$299 \times 299 \times 3$
conv	$3 \times 3 / 1$	$149 \times 149 \times 32$
conv padded	$3 \times 3 / 1$	$147 \times 147 \times 32$
pool	$3 \times 3 / 2$	$147 \times 147 \times 64$
conv	$3 \times 3 / 1$	$73 \times 73 \times 64$
conv	$3 \times 3 / 2$	$71 \times 71 \times 80$
conv	$3 \times 3 / 1$	$35 \times 35 \times 192$
$3 \times$ Inception	As in figure 4	$35 \times 35 \times 288$
$5 \times$ Inception	As in figure 5	$17 \times 17 \times 768$
$2 \times$ Inception	As in figure 6	$8 \times 8 \times 1280$
pool	8×8	$8 \times 8 \times 2048$
linear	logits	$1 \times 1 \times 2048$
softmax	classifier	$1 \times 1 \times 1000$

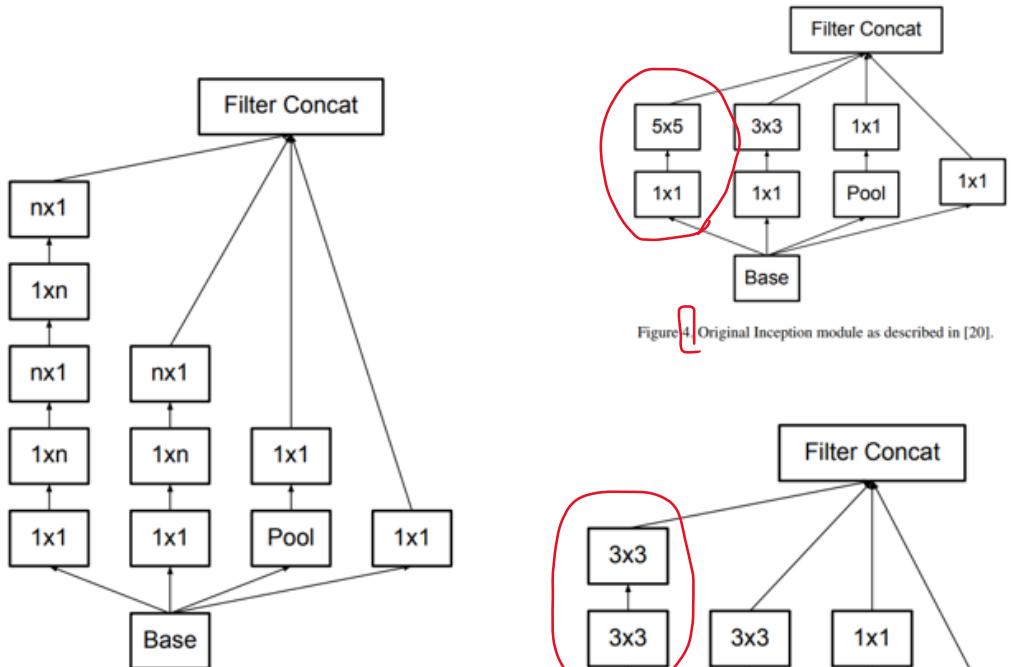


Figure 6. Inception modules after the factorization of the $n \times n$ convolutions. In our proposed architecture, we chose $n = 7$ for the 17×17 grid. (The filter sizes are picked using principle 3)

Figure 4. Original Inception module as described in [20].

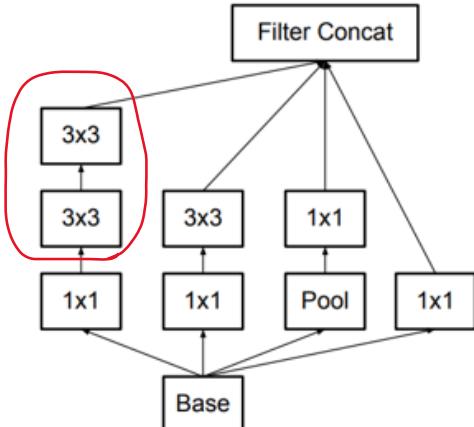


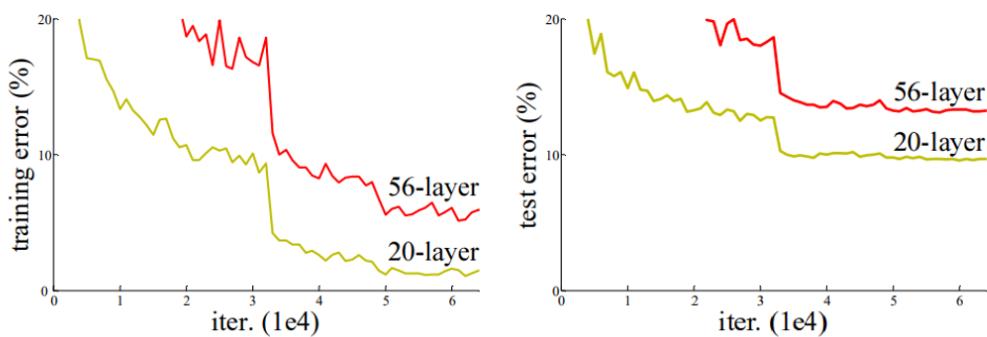
Figure 5. Inception modules where each 5×5 convolution is replaced by two 3×3 convolution, as suggested by principle 3 of Section 2.

ResNet – can't we just go deeper & deeper

הבסיס שעומד מאחורי הרשת הגדת היא התובנה משני הגפים הבאים:

- Notice 2 phenomena:

- Test error significantly higher than training, but also...
- Training error for 56-layer network is also higher than that of 20-layer net!



אם אנחנו מסתכלים על training error - test error אז נראה שNETWORKS שרטות עוקבות יותר יצילחו להציג שגיאה הולך וקטן ככל שהן יותר עמוקות, מכיוון שיש לנו דרגות חופש הולכת ומעלה.

זה לא היה מפתיע אותנו אם הינו מקבלים LOSS גבולה יותר או שגיאה גדולה יותר עברו נתוני הטסט שלנו והסיבה לכך היא שיכול להיות שאחנו במצב של Overfitting.

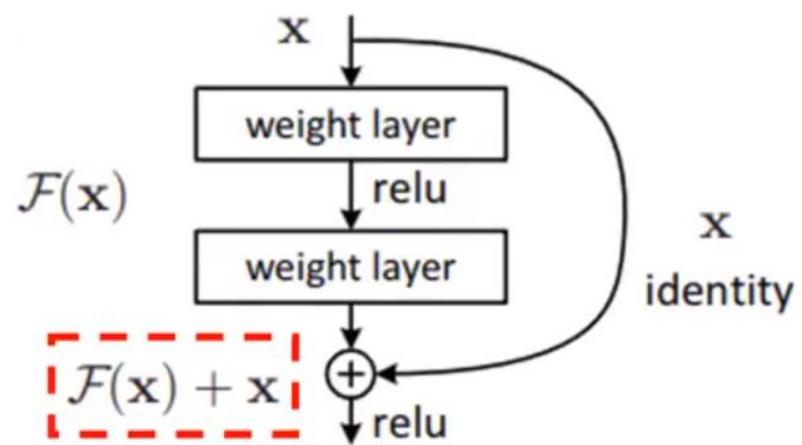
לכן רשות עם יותר שכבות תצליך פחות ביצוע המשימה, אך התובנה המעניינת, שאחנו מגיעים לתוצאות פחות טובות גם בשגיאת האימון, ואם אנחנו מגיעים לתוצאות פחות טובות בשגיאת האימון אז זה לא אומר שאחנו במצב של Underfitting, אלא במצב של Overfitting.

נזכיר שאחנו מגיעים במצב של Underfitting כאשר אנחנו לא מצליחים לכנס את תהליכי האופטימיזציה, יש לנו הרבה פרמטרים ודרגות חופש ואחנו עדין לא מצליחים לכנס את התהליכי ובאופן פשוט יותר, המודל לא מצליח לשנן אפילו דוגמאות שהוא כבר ראה.

از בעצם אנחנו מבינים שלא מדובר בעיה אופטימיזציה, لكن הנחת העבודה הייתה שבמקום שננסה למדוד כל שכבה ברשות בהינתן שהאחרות מתכנסות אנחנו נרצה למדוד כל שכבה שאר השכבות היי מעבירות את הטעות כמו שהיא (learn the residual error).

איך געשה את זה?

כותבי המאמר מציעים בлок בסיסי שהוא קצר שונה. אם עד עכשיו דיברנו על שרשור של שכבות, או שרשור של קלטים מהשכבה הקודמת (כמו ב-Inception), אז הפעם נבצע חיבור של איזשהו בлок (במקרה שלנו קונבולוציות אבל באותה מידת זה יכול להיות חיבור של מרכיבים אחרים) שאמור ללמידה איזה שהוא יחד עם פונקציית הגדת. כלומר או שלמדתי משהו שמצויץ את LOSS הכלול, או לחלוfin מה שאני לומד זה רק את ההפרש.

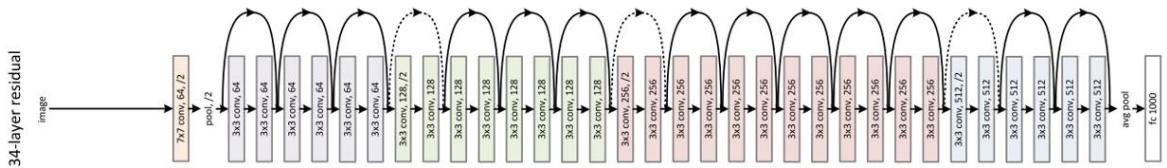


בהקשר לתמונה, יכול להיות ש- X כבר מותאם לפולט הרצוי, אבל יכול להיות שלא ואז אנחנו ננסה את ההפרש בין X לבין הפולט הרצוי. (קצת מזכיר האלמנט של boosting שאנו חנכו מנסים ללמידה את השאריות של איזשהו loss אחר שה חישבנו כבר).

אם רשת רגילה נראית ככה:

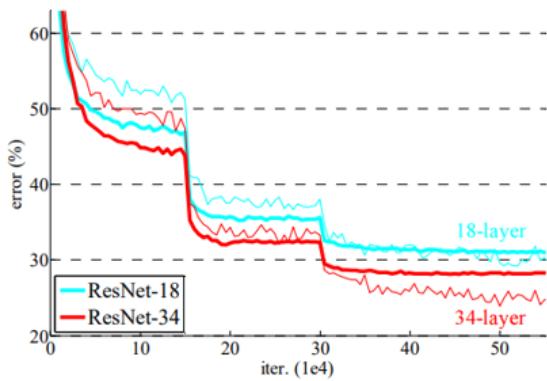
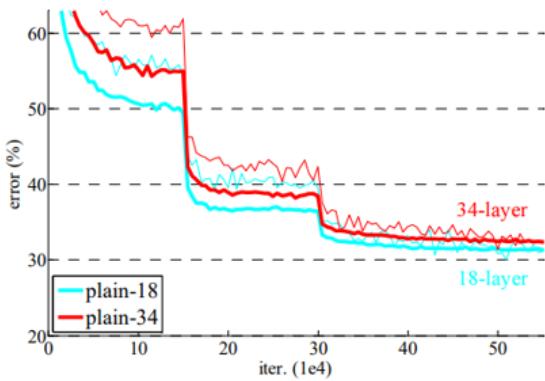


از רשת מהסוג של ResNet תיראה ככה:



ובעצם כפי שהסביר, במידה שיש שכבה שאנו לא לומד ממנה כלום אז אני פשוט מעביר את המידע אליו כפי שהוא, דבר זה יכול מאד להקל בשלב הלמידה ויהי יותר קל לשכבות הראשונות להתכנס ביחס לרשת ללא האופציה לעשות skip בין השכבות.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112			7×7, 64, stride 2		
				3×3 max pool, stride 2		
conv2_x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1			average pool, 1000-d fc, softmax		
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9



Batch Normalization

קצת על Batch Normalization, מה שאנחנו הזכינו כבר זה שתהליכי רשותה שהנתונים שאנו מעבירים (נתוני הבסיס) מנוורמלים, דבר זה תורם להתקנות מהירה.

הweeney ב- Batch Normalization הוא שיכל להיות מאוד מהקלט שלו הגיע מנוורמל, ואחרי הפעולות שביצענו בשלבים המוקדמים עד אליו השלב אמצעי הבאנו אותו שוב לUMB שהוא בעצם לא מנוורמל.

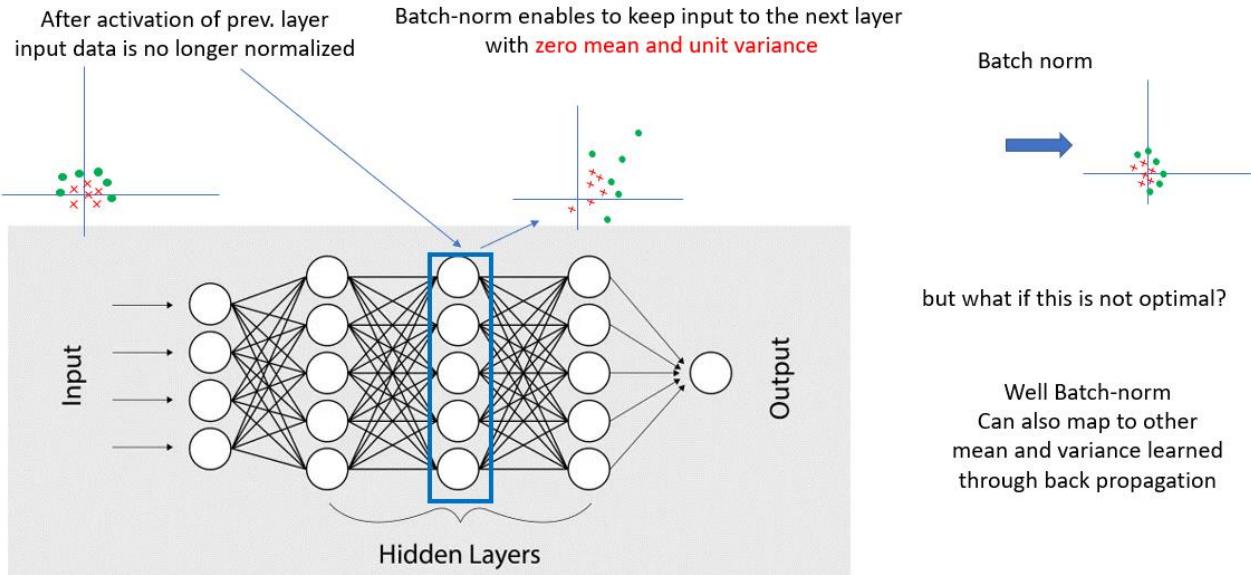
לא נרצה להסתפק בנוורמל שיביא אותנו ל – unit variance and zero mean, אנחנו נרצה את האפשרות שניקח את כל הדבר הזה לנוקודה אחרת אחרי ביצוע אותו נורמל, להסיט את כל הנקודות האלה לנוקודה אחרת שהיא אופטימלית.

איך עושים את זה?

המנגנון של Batch Normalization אומר את הדבר הבא, עבור כל מיני באץ' נחשב את הממוצע ואת השונות, אנחנו נורמל ולאחר כך נשתמש בטרנספורמציה לינארית באמצעות פרמטרים גמא ובטא ואת התוצר תהיה אותה טרנספורמציה מעל הקולט המנוורמל.

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_1 \dots m\}$;	
Parameters to be learned: γ, β	
Output: $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$	
$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i$	// mini-batch mean
$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2$	// mini-batch variance
$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$	// normalize
$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i)$	// scale and shift

בשביל למנוע שבסכל ריצה תהיה תוצאה שונה על אותה דוגמה כתלות בין הדוגמאות האחרות שראינו באותו batch, שמדובר זה בדבר שהוא פחות רצוי מבחיננו, אך נשתמש בתוחלת והשונות הן הממוצע על פני כל batches שהוא לנו.



ML/DL strategy

- Many possible next steps
 - Add more data (quantity, diversity, specific sub-classes)
 - Clean existing data (remove outliers,
 - Augment existing data (transformations, random noise, specific noise, cyclic translation)
 - Use different metric/loss
 - Use a different validation method
 - Change architecture (width, depth, connectivity)
 - Different type of network (Conv / Fully Connected / RNN etc.)
 - Increase / decrease regularization
 - Use different initialization

Choose **well** to **advance quickly!**
Or choose **poorly** to **waste weeks or even months**

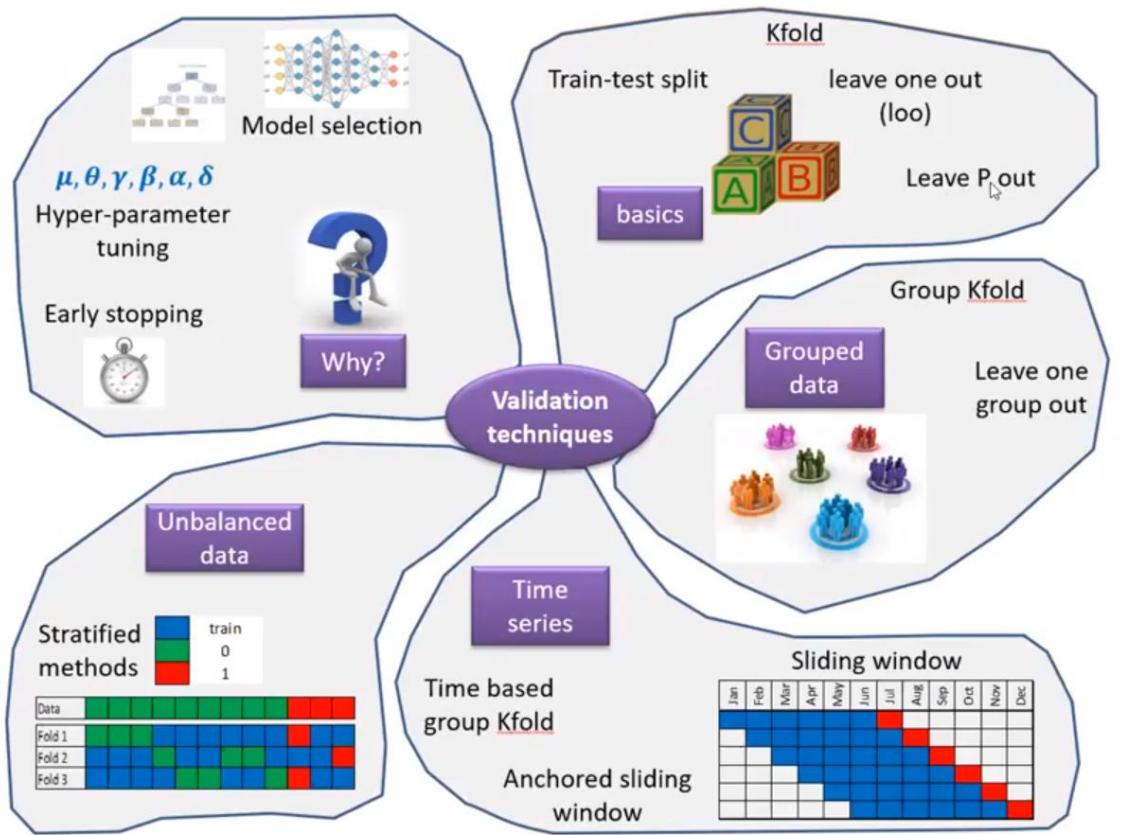
בחירה נכונה יכולה להיות מאד קריטית לגבי ההצלחה שלנו בתחום הלמידה.

Validation strategy

הבסיס של רעיון הווילדייצה הוא להכרייז מראש מה התוצאות שאנו מוצאים לקבל בשימוש באלגוריתם מסוון, כלומר שאימנו כבר את הרשות או את המודל האخر שבו אנחנו משתמשים, אנחנו נרצה בעצם להיות מסוגלים לתת תוצאה רציה.

כל האצעה היכי חשוב בהקשר של ווילדייצה הוא שווילדייצה אמורה לשקף את הנתונים שאנו מוצאים לקבל, אם אנחנו מגדירים משימה של פרדיקציה ולחזות איזה אירוע עתידי, אז אנחנו לא יכולים להתבסס כפיצרים על שום דבר שלא יהיה ידוע מראש בזמן ביצוע הפרדיקציה, מה זה אומר? שאנו לא יכולים להתבסס על נתונים עתידיים לדוג' איזה מדידה שהתבצעה חמיש דקות לפני הזמן שאנו מנסים לחזות אם הזמן שאנו מנסים לחזות הוא 24 שעות קדימה.

שיטת הוילדייצה שלנו צריכה לתמוך בזה, אנחנו לא נוכל להסתכל על מהهو שלא יהיה זמן בשעת ביצוע הפרדיקציה, לעומת זאת אם ניקח פיצר של יום בשבוע אז גם עבר פרדיקציה של מספר ימים קדימה נדע להגיד מה היום בשבועו הקרוב לאותו פרק זמן שאנו מנסים לחזות.



- Train-test split



- Shuffle split



- Kfold



Train-test split – אני מחלק באופן אקראי בין הדוגמאות שישמשו אותי עבור training ועבור validation. (החלוקת לא חיבת להיות סדרתית וכן יכולה להיות מעורבתת)

Shuffle split – ביצוע חוזר K פעמים של Train-test split. שונות מfold K בכך שאנו משתמשים בshuffle split אנחנו יכולים לבחור נגיד 70% שליר עבר האימון ו30% עבר הווילידציה, וכן כל שימוש פועלה זאת 10 פעמים שונות כי פועלה זאת היא פעולה עם החזרה.

Kfold – אנחנו מדברים על בחירה ללא החזרה ואז K פעמים אנחנו צריכים לכסות בסך הכל את כל הדוגמאות, כלומר כל דוגמה תשמש פעם אחת עבר ווילידציה ו-1-K פעמים עבר אימון.

- Leave one out



- Group K fold



(oo) Leave one out – בשיטה זו זה מקרה פרטי של Kfold שווה במספר הדוגמאות, כלומר כל פעם הווילידציה היא דוגמה בודדת וכל שאר הנתונים משמשים עבר האימון. מתי נרצה להשתמש במצב הזה? כאשר אני רוצה לאמן כמות גדולה של מודלים, ואני רוצה לשים לב לשינויים הקטנים ביניהם ולהיות מאד בטוח בבחירה שלי, אם בבחירה בסוג המודל או בהיפר פרמטרים, אך יש לה מחר – אימון רב של מודלים יכול לקחת לא מעט זמן אז בכל זאת מתי נשימוש בזה? כאשר יש לנו מעט מאוד דוגמאות בסך הכל.

– Group Kfold

שאלה: למה שנרצה להשתמש בgroup ?
תשובה: יש הנחה שאנו מחלקים לחוקים train | test יש הנחה שנתקראת ID, שאין תלות בין הנתונים השונים. אך יכול להיות שהנתונים תלויים זה בזה והם לא מתפלגים בצורה אחידה. לדוג' לדוגמה דגימות מאנשים שונים, כאשר כל אחד מתנהג בצורה שונה שונה אבל בין הדגימות כן מתנהג בצורה דומה.

למשל, בשיטה כזו אנחנו יכולים לבצע אותה חלוקה בהקשר של קבוצה. Group K fold – הפרדה מבוססת קבוצות, נחזיק אנשים שונים בסט האימון ובסט הווילידציה.

**כשיש לנו נתונים תלויי זמן הנתונים חייבים להיות סידרטים, כי יש חשיבות גם לזמןם לפני ואחרי הנקודות שמייצגות את הדוגמאות של'.

במקרה זה נשמש במנגןון ולידzieה שנקרא **Sliding window**

- Sliding window



דרך נוספת נקראת **Anchored sliding window** בעצם כל הזמן נסתכל על הנתונים מאותה נקודת התחלתית וכל פעם נוסיף נתונים נוספים.

- Anchored sliding window



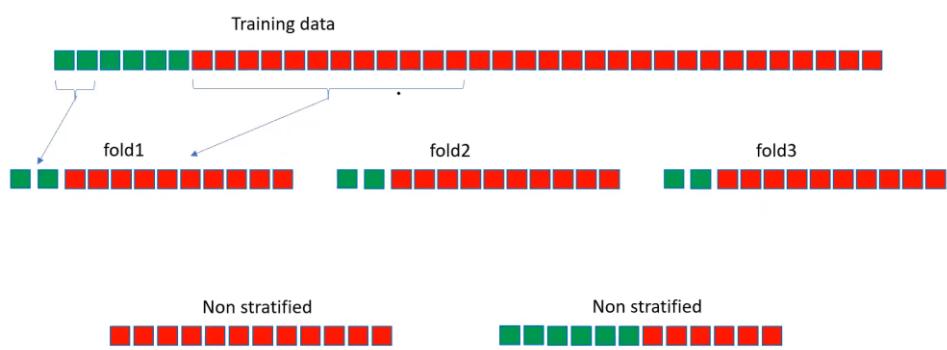
Validation for unbalanced data

קורה כישיש לו מחלוקת שמייצגת יתר על המידה או באופן לא מספיק. במצב זה נרצה לדגם אחוז זהה מכל אחת מן האפשרויות.



בעצם הבעיה שלנו שיכל להיות לנו מצב שאין לנו בכלל דוגמאות מהקהלאוס הפחות מיוצג נתונים או להפוך שנקלבל אובר יציג של אותו קלאוס שהוא לא כל כך שכיח, אז החוסר יכולות שלנו להתאים לקלאוס הפחות שכיח תשפייע באופן גבוה מדי על התוצאה הכלולת שלנו.

לעומת זאת מה שכנ נוכל לעשות זה לחלק באחוזים די זהה כל אחד מהמשתנים, שיטה זאת נקראת **.stratification**.

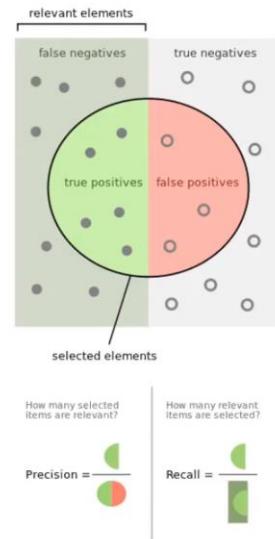


Metrics

יש מטריקות שמתאימות למשימות שונות, מה חשוב שלא תמיד נרצה למקסם או למזער, מטריקות של הפסד נרצה למזער ומטריקות של דיווק נרצה למקסם, אבל לא תמיד זה מה שנרצה מטריקה מסוימת. לעיתים נרצה להיות מה שנקרא "good enough" ובהינתן זאת אנחנו מבדלים בין satisficing and optimizing metrics.

המטריקות שהן יכולות להיות שמספיק שמצאנו X דוגמאות ובהינתן שמצאנו לפחות X דוגמאות, אנחנו רצים Precision מאוד גבוהה.

- Use **many metrics** to have **better insights** on our results
auc, f1-score, precision, recall, map@k etc. for classification
and RMSE, MAE, RMSLE, MAPE etc. for regression
- select a **single metric** to **prioritize** different **methods**
- Differentiate **satisficing** and **optimizing** metrics



Change our metric or validation method

מתי נרצה בעצם לשנות את המטריקה שאיתה אנחנו מודדים?

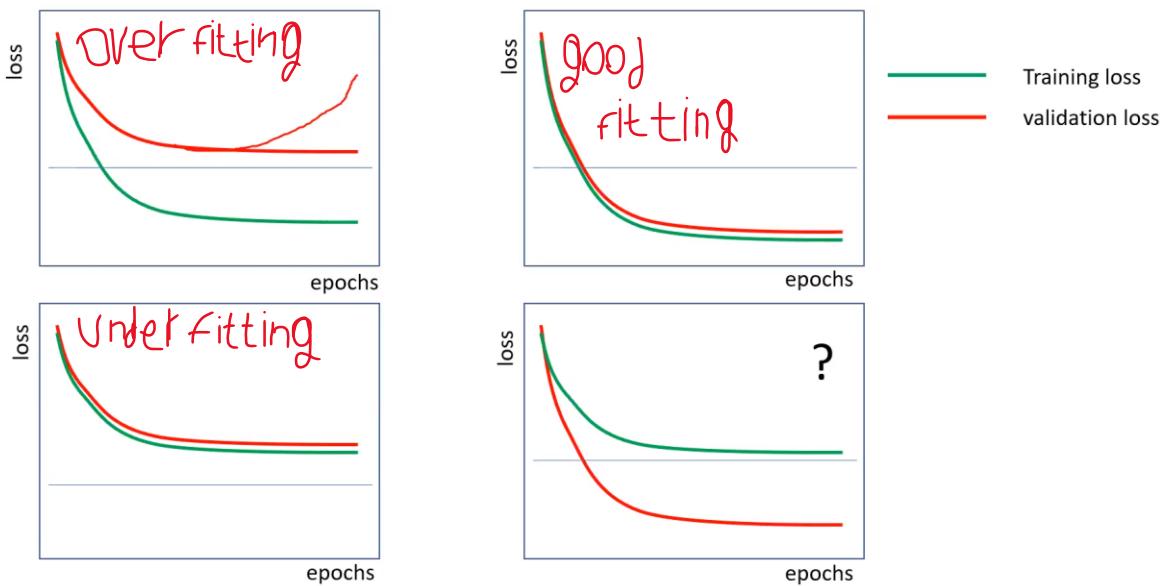
מקרה ראשון יכול להיות שהפתרונות שאנחנו מקבלים הן שנות מאיזשה הנחת בסיס שהנחנו בתחילת התהילה, למשל הנחנו שהנתונים לא תלויים בזמן ואז גילנו שהוא בוחנו בו קודם.

מקרה נוסף יכול להיות כשאנו מקבלים תוצאות טובות מדי על סט הולידציה שלנו. לדוג' לפעמים יכול לקרות שסט הולידציה שלנו בנויות מדוגמאות פשוטות מדי, ובשלב זה אנחנו עוד לא ידעים שיש מרכיבות שונה בין הדוגמאות שלנו.

מקרה אחרון הוא בחירה לא נcona של satisficing and optimizing metrics, לפעמים נרצה ליצור איזשה הפרדה של מטריקות שהיא לא נcona ונגלה את זה בבדיקה.

מלבד מצבים אלה לא נרצה לשנות את שיטת הולידציה.

Overfitting, Underfitting and...



בצד ימין למטה אנו רואים דוג' למצב לא שכיח, תופעה שבנתוני האימון אנחנו לא מאוד טובים אבל בנתוני הולידציה אנחנו כן יכולים מנייבים תוצאות טובות, דבר זה יכול לקרות בעקבות שהבחירה שלנו לנוטוי הולידציה הם לא בחירה טובה, אולי הם לא מספיק מרכיבים ושהם פשוטים מדי ונתוני האימון הם יותר מרכיבים ביחס לנוטוי הולידציה.

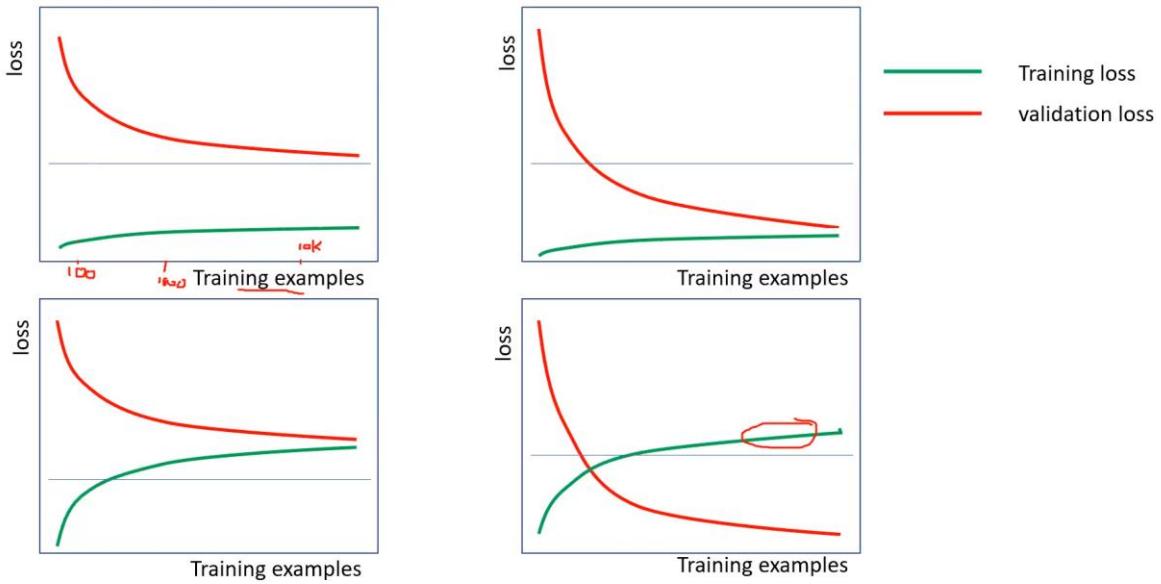
זהינו שיש לנו תופעה של **Overfitting**, אז מה עושים?

- נגדיל את סט האימון
- נשתמש באוגומנטציה (באופן זהיר כמובן, נזכר בדוגמא של 6 ו 9)
- לצמצם את מרכיבות המודל באופן שלא ניתן שינון להגדיל את הריגוליזציה
- לשימוש ב- **Dropout**
- לעזר את האימון בשלב מוקדם לפני שהתופעה של Overfitting מתרחשת
- לשנות את הארכיטקטורה של המודל
- ושימוש ב- **Bagging**

זהינו שיש לנו תופעה של **Underfitting**, אז מה עושים?

- . Overfitting.
- בגדול ההופכי של מה שעשינו במצב של Overfitting.
- להגדיל את מרכיבות המודל
- להקטין את הריגוליזציה
- לשנות את הארכיטקטורה של המודל
- שימוש ב- **Boosting**

Is adding additional data a good next step?



ציר ה-X מתראר את מספר הדוגמאות של האימון, כמוות הדוגמאות של האימון יכולה להמחיש שכל פעם אנחנו משתמשים בחלק שהולך וגדל מהdatasetו, לדוג' נתחיל ב100 דוגמאות, 1000 דוגמאות ו10,000 דוגמאות. זה הגיוני שאם נגדיל את מספר הדוגמאות האימון שלנו אז גם loss training יגדל כי בהתחלה יש מגוון של נתונים שקל להתאים ל100 דוגמאות אבל הרבה יותר קשה להתאים ל10,000 דוגמאות.

שמאל למעלה – מצב שכמות הדוגמאות לא עוזרת לקבל תוצאה יותר טובה בולדיצה.
ימין למעלה – נוכל לקבל מצב שהוא כן יעזור לשפר את יכולת הכללה שלנו.

שמאל למטה – נוכל לקבל מצב שלא רק שלא יהיה יותר טובים, אלא לא מספיק מתאים את העולם במצב ההתחלתי, אולי בהתחלה נצליח לשנן אבל ככל שנוטיף יותר דוגמאות אנחנו גורעים מיכולת הייצוג של האימון.

ימין למטה – נוכל לקבל מצב שהוא פחות נפוץ אבל קורה, שאנחנו מצלחים להכלייל אבל אנחנו מוסיפים דוגמאות ברמת מורכבות גבוהה יותר שאנו לא מצלחים להגיע לתוצאה מאוד טובה באימון.

השאיפה הוא להגיע לגרף מרהסוג שבו רואים **בימין למעלה**.

Suggested process to train DL networks

- נסה להבין איזה סוג רשות נדרש לעבוד עבור הבעה הנתונה
- החלטת מה יהיה validation טוב לבעיה מהסוג זהה
- יוצרים פתרון ראשוני ולהריץ אותו
- היקן שניתן, השתמש בארכיטקטורות pretrained כנקודות התחלה
- נסה בהתחילה להציג למצב של Overfit על נתונים האימון -> המודל מרכיב מסוים
- הוסיף רגולציה / נשירה / השתמש במודל פשוט יותר

Debugging

לא תמיד נקבל את התוצאה הרצiosa על ההתחלה, ולכן מה נctrיך לעשות?

- הדפס נתונים קלט ופלט
- הדפס שונות נתונים
- חפש חריגים וערכים לא מתאימים
- הדפס פרדיקציות של שכבות (לא כל כך טוב עבור ארכיטקטורות גדולות)

Beware of:

- סחף קונספט??
- הטיה פוטנציאלית שאינה מיוצגת נתונים
- התאמת יתר לסט אימונות

הרצאה 5:

Embedding

השיטה שבאמצעותה ננסה להבין מה זה embedding זה באמצעות דירוגים מסוימים עבור סרטים של משתמשים מסוימים ונראה לייצר איזה מערכת דירוג של סרטים עבור משתמשים שלא ראו את אותו הסרט, או מנסה לשער מה הדירוג שהמשתמש היה נותן לו והיה רואה את אותו הסרט.

Raw data form:

userId	movieId	rating	timestamp
73	1097	4	1255504951
561	924	3.5	1172695223
157	260	3.5	1291598691
358	1210	5	957481884
130	316	2	1138999234
580	1196	4	1220561546
544	2918	5	1435787004
213	1200	3	1462634054
176	2571	4.5	1340714691
481	4886	4.5	1437002227
423	1270	3	1353700413
607	364	3	1144791031
105	4963	3	1086093708
165	500	3	1111482850
665	32	4	995233393
547	2997	5	974779039
292	780	4.5	1140051052
134	231	2.5	1361245725
665	597	5	992920864

After performing cross tabulation																
*	most frequent users and movies,															
*	with some random erasing for our educational purposes															
userId	movieId	27	49	57	72	79	89	92	99	143	179	180	197	402	417	505
	14	3	5	1	3	4	4	5	2	5	5	4	5	5	2	5
	29	5	5	5	4	5	4	4	5	4	4	5	5	3	4	5
	72	4	5	5	4	5	3	4.5	5	4.5	5	5	5	4.5	5	4
	211	5	4	4	3	5	3	4	4.5	4	3	3	5	3	4	3
	212	2.5		2	5	4	2.5		5	5	3	3	4	3	2	
	293	3		4	4	4	3		3	4	4	4.5	4	4.5	4	
	310	3	3	5	4.5	5	4.5	2	4.5	4	3	4.5	4.5	4	3	4
	379	5	5	5	4		4	5	4	4	4		3	5	4	4
	451	4	5	4	5	4	4	5	5	4	4	4	4	2	3.5	5
	467	3	3.5	3	2.5			3	3.5	3.5	3	3.5	3	3	4	4
	508	5	5	4	3	5	2	4	4	5	5	5	3	4.5	3	4.5
	546		5	2	3	5		5	5		2.5	2	3.5	3.5	3.5	5
	563	1	5	3	5	4	5	5		2	5	5	3	3	4	5
	579	4.5	4.5	3.5	3	4	4.5	4	4	4	4	3.5	3	4.5	4	4.5
	623		5	3	3		3	5		5	5	5	2	5	4	

בשלב ראשון ננסה למדل את הבעה בצורה הבאה, ניתן למשתמש ייצוג חדש שנatatחל בצורה אקרואית ובאופן דומה נייצג כל סרט שגם אותו נייצג באופן אקרואי. בצורה שירוטית נבחר בדוג' ייצוג של 5 ממדים לאותו ייצוג חדש.

בנקודות הצלבה בין המשתמש לסרט נשים את הדוט פרודקט מהייצוג של הסרט ושל המשתמש.

NB: These are initialized to random numbers
Then we use Solver to optimize them
with gradient descent

0.42	0.48	0.12	0.62	0.57	0.74	0.37	0.66	0.92	0.83	0.51	0.50	0.92	0.01	0.83						
0.60	0.34	0.95	0.63	0.19	0.64	0.60	0.92	0.05	0.07	0.17	0.47	0.41	0.69	0.83						
0.15	0.75	0.01	0.51	0.08	0.61	0.10	0.67	0.41	0.63	0.35	0.47	0.97	0.16	0.05						
0.06	0.07	0.94	0.53	0.21	0.09	0.83	0.06	0.67	0.35	0.65	0.30	0.88	0.56	0.05						
0.61	0.52	0.39	0.97	0.69	0.05	0.10	0.03	0.03	0.84	0.69	0.45	0.92	0.63	0.62						
userId	movieId	27	49	57	72	79	89	92	99	143	179	180	197	402	417	505				
0.82	0.85	0.14	0.78	0.70	14	1.34	1.21	1.90	2.21	1.29	1.34	1.55	1.48	1.40	1.68	1.61	1.42	2.56	1.48	1.86
0.94	0.60	0.54	0.76	0.48	29	1.17	1.37	1.58	2.11	1.19	1.51	1.44	1.59	1.64	1.83	1.60	1.45	2.74	1.23	1.64
0.72	0.14	0.96	0.72	0.35	72	0.78	1.35	1.04	1.75	0.91	1.29	1.08	1.30	1.56	1.76	1.45	1.25	2.60	0.88	1.01
0.58	0.08	0.72	0.29	0.90	211	0.96	1.34	0.77	1.80	1.08	0.99	0.66	0.98	1.05	0.00	1.37	1.15	2.33	0.90	0.00
0.91	0.10	0.01	0.26	0.46	212	0.74	0.00	0.63	1.21	0.00	0.79	0.66	0.00	1.03	1.24	0.97	0.79	1.53	0.51	1.14
0.03	0.74	0.02	0.39	0.04	293	0.50	0.00	1.09	0.74	0.27	0.55	0.00	0.74	0.34	0.26	0.43	0.50	0.72	0.75	0.00
0.60	0.55	0.27	0.12	0.12	310	0.70	0.75	0.75	1.04	0.57	0.99	0.69	1.10	0.78	0.85	0.66	0.77	1.25	0.57	1.05
0.03	0.54	0.77	0.09	0.01	379	0.46	0.80	0.61	0.82	0.00	0.86	0.49	1.05	0.44	0.60	0.00	0.66	1.09	0.55	0.53
0.70	0.55	0.95	0.84	0.90	451	1.36	1.77	1.75	2.59	1.38	1.57	1.47	1.68	1.66	2.27	1.96	1.71	3.35	1.58	1.69
0.47	0.67	0.44	0.51	0.90	467	1.24	1.30	1.52	2.09	0.00	0.00	1.14	1.28	1.02	1.65	1.47	1.31	2.41	1.39	1.55
0.01	0.25	0.18	0.39	0.74	508	0.65	0.64	0.88	1.18	0.66	0.34	0.57	0.40	0.38	0.90	0.88	0.65	1.30	0.88	0.70
0.20	0.12	0.32	0.75	0.36	546	0.00	0.62	0.98	1.11	0.57	0.00	0.83	0.51	0.00	0.94	0.98	0.70	1.54	0.78	0.54
0.78	0.49	0.24	0.65	0.51	563	1.00	1.03	1.36	1.75	1.05	1.12	1.20	0.00	1.29	1.49	1.35	1.16	2.19	1.06	1.41
0.71	0.41	0.34	0.30	0.89	579	1.16	1.23	1.11	1.90	1.19	1.07	0.89	1.13	1.05	1.69	1.37	1.20	2.23	1.07	1.52
0.83	0.29	0.34	0.85	0.17	623	0.00	0.91	1.25	1.49	0.00	1.10	1.24	0.00	1.50	1.37	1.27	1.05	2.12	0.84	1.10

אנחנו נשאף להביא את המספרים האלה למספרים מהטבלה ראשונה, ובהתאם לשנות את המשקלים המאותחלים שקבענו על מנת להגיע לתוצאה הרצויה (בדוג' המסוונת באדום באתחול הראשוני הוא 1.34 לאחר דוט פרודקט, והיינו רוצים שהמספר יהיה קרוב ככל האפשר ל³)

איife שאין מספר נרצה לא להימדד בשלב זה.

באופן לא מפתיע מה שנרצה לעשות זה לחשב את הגрадיאנט דיסנט בין הנגזרת החלקית של LOSS שלנו שיכול להיות ריבוע ההפרש או MSE בין מה שנקבל בתוצאה הרצויה לבין התוצאה שאנו מקבלים בדוט פרודקט בטבלה השנייה.

לאחר תהליך אופטימייזציה ניתן לראות שנגיע לטבלה הבאה

After optimizing using gradient descent
we achieve the following results:

	-0.49	1.67	-0.60	1.55	0.86	2.43	2.07	1.04	0.04	1.33	0.65	1.03	0.63	0.79	1.79					
	0.66	0.81	1.33	1.02	1.07	1.00	0.66	1.52	0.17	0.37	0.83	0.67	0.40	0.97	0.96					
	1.98	-1.36	3.24	1.90	2.02	0.42	-2.22	-0.12	2.11	0.37	1.03	0.36	2.71	0.46	-1.01					
	-0.61	0.67	1.04	1.50	-0.41	0.03	0.50	0.20	2.02	2.30	2.64	1.95	-0.79	1.84	0.16					
	2.65	2.57	-0.40	-1.67	1.33	-0.16	2.76	1.50	0.45	0.17	-0.38	0.20	1.42	0.05	2.25					
userId		27	49	57	72	79	89	92	99	143	179	180	197	402	417	505				
2.15	-1.54	1.14	1.09	1.39	14	3.22	5.07	0.94	3.24	3.90	3.99	5.29	2.06	5.04	5.45	3.62	4.01	4.95	2.80	4.52
0.59	2.06	0.64	0.95	0.99	29	4.40	4.96	5.07	4.00	4.95	3.62	4.36	5.33	4.10	4.13	4.88	4.28	3.60	4.55	4.77
0.74	1.44	0.82	1.20	1.11	72	4.43	4.94	4.96	4.14	4.83	3.44	4.32	4.76	4.95	4.77	5.28	4.61	3.90	4.63	4.57
0.80	1.02	0.97	0.49	1.21	211	5.11	4.28	4.04	2.83	5.14	3.18	3.77	4.17	3.79	0.00	3.19	3.07	4.86	3.03	0.00
1.92	-1.27	1.34	0.88	0.61	212	1.97	0.00	2.18	4.54	0.00	3.91	2.29	0.00	4.74	4.72	3.66	3.47	4.51	2.56	2.38
1.00	0.55	0.99	0.94	0.76	293	3.28	0.00	4.02	4.13	4.08	3.30	0.00	3.08	4.46	4.18	4.30	3.74	3.87	3.54	0.00
0.94	1.66	0.94	0.53	0.46	310	3.38	3.17	5.06	4.98	4.88	4.29	2.49	4.18	3.59	3.52	4.18	3.56	4.03	3.79	3.46
1.04	1.16	1.02	0.66	1.16	379	4.95	4.70	4.47	3.82	0.00	3.96	4.18	4.59	4.26	3.91	0.00	3.76	5.02	3.70	4.67
0.85	1.84	0.43	0.87	0.81	451	3.28	5.00	3.93	3.98	4.31	3.99	4.71	5.03	3.39	4.11	4.51	4.14	2.92	4.30	4.84
0.54	1.07	0.53	0.79	0.83	467	3.22	3.71	3.30	2.72	0.00	0.00	3.34	3.52	3.29	3.26	3.54	3.17	2.77	3.19	3.47
0.71	0.48	1.00	1.15	1.48	508	5.17	4.78	4.06	2.76	4.65	2.44	4.23	3.80	5.20	4.38	4.35	3.97	4.55	3.68	4.25
1.29	1.64	0.36	0.05	0.79	546	0.00	5.04	2.34	3.14	4.63	0.00	5.14	4.97	0.00	2.71	2.41	2.83	3.53	2.92	5.30
1.56	1.59	0.32	0.66	0.36	563	1.47	4.81	2.78	5.05	3.90	5.47	4.88	0.00	2.51	4.36	4.26	4.16	2.48	4.15	4.91
1.14	0.77	0.88	0.66	1.08	579	4.16	4.56	3.44	3.41	4.75	3.77	4.25	4.01	3.85	3.83	3.61	3.53	4.42	3.32	4.45
0.75	0.75	0.39	1.60	1.06	623	0.00	5.11	3.06	3.30	0.00	2.61	4.91	0.00	4.69	5.28	5.33	4.76	2.07	4.50	4.30

RMSE after optimization 0.39

דבר חשוב שציריך לשים לב זה שיש משתמשים שבאופן אישי יותר מפגנים בדירוג סרטים, לעומת משתמשים שפחות מפגנים בדירוג הסרטים וזה לא אומר בהכרח שהסרט טוב או לא, באותו אופן יש סרטים שהדירוג שלהם יותר גבוה וייתר נמור מסרטים אחרים במשמעותם.

דבר זה יכול להשפיע על מערכת המלצתה שלנו ויכולו לגרום למצב שנדרג עבור משתמש ציון מסוים לסרט ובגלל שאט הסרט באופן כללי יש לו דירוג נמור לאוטו משתמש "מגן" שנקרא לשערך את הציון שהוא יתן אנחנו עצמנו נזהה שהוא יתן ציון נמור לסרט למורות שהוא לא בהכרח נכון.

פתרון? נסיף הטיה (bias) לכל משתמש שיכול לאזן את הציון שנছזה עבור המשתמש.

בצם אפשר להסתכל על זה כי צג וקטורי של העדפות של אותו משתמש לגבי סרטים (וכמובן שאפשר לראות אותו דבר גם מבחינת הסרטים).

The received weights of the embedding can be perceived as a hash table
in which the key is the ID and the values are corresponding vectorized representations

key	Vectorized representation
14	0.11 2.04 -0.95 0.92 0.99 1.24
29	0.85 0.40 1.56 0.58 0.92 0.75
72	1.90 0.00 0.65 0.28 1.13 0.81
211	0.82 0.53 0.82 0.69 0.24 1.03
212	1.32 2.00 -1.04 1.07 -0.25 -0.09
293	1.88 0.40 0.12 0.26 0.42 0.36
310	1.98 0.74 0.92 0.59 -0.36 -0.20
379	1.71 0.21 1.10 0.17 0.39 0.91
451	-0.22 1.09 1.71 0.79 0.97 0.71
467	-0.06 0.54 0.94 0.54 0.88 0.84
508	1.36 -0.09 0.21 0.26 1.22 1.40
546	-1.53 2.38 1.16 1.91 0.31 1.24
563	0.87 0.97 1.25 -0.42 0.72 0.39
579	0.50 1.03 0.82 0.71 0.43 0.91
623	-0.42 0.80 0.68 0.93 2.06 1.22

איך זה embedding?

- ייצוג חדש של הבעייה
- ייצוג חדש של הנטיונים

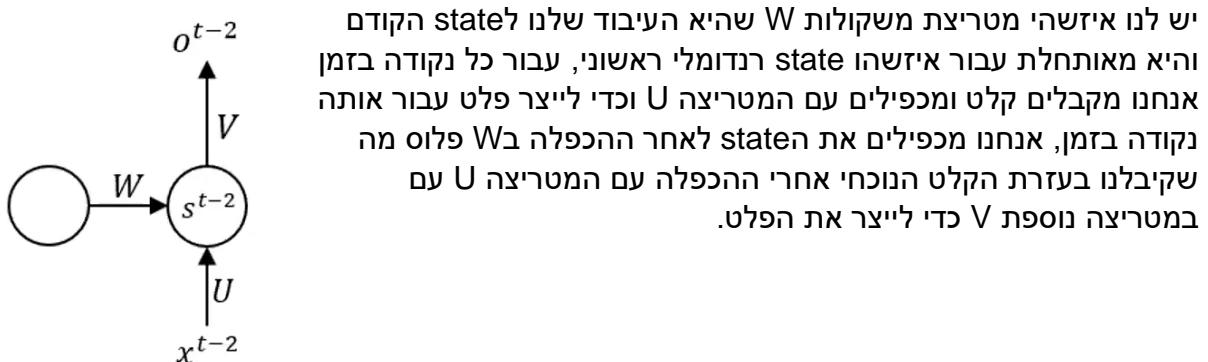
אנחנו בעצם לומדים ייצוג חדש של אלמנט מסוים מתוך המידע או יותר לאור הבעייה הנתונה, ככלומר אנחנו לא לומדים תכונות גובהה ומשקל של המשתמש אלא לומדים ייצוג שיעזר לתאר יותר טוב משתמש לאור בעיית דירוג הסרטים.

RNN – Recurrent neural networks

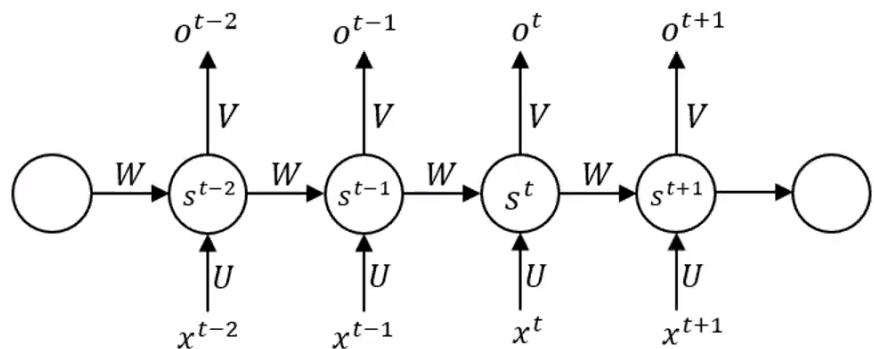
אינטואיציה – למה אנחנו צריכים עוד סוג של מודל?

- נרצה לפעמים להתייחס לאורכים משתנים של סיגナル או sequence, למשל אנחנו יכולים לקבל איזשהו רצף קצר וארוך ונרצה למדל את שניהם בצורה דומה.
- לצד זה נרצה לפעמים לשמר איזה מנגן של זיכרון או state מצב עדכני של הרשות, למשל אם נחשוב על ביטוי מתמטי מסוים, נרצה להגיד אם הביטוי הוא וולידי או לא וולידי בהתאם למספר הסוגרים שפתחנו. בעצם נרצה לשמר איזשהו זיכרון של כמה סוגרים פתוחים ומאייזה סוג היה לנו בכל שלב.
- היכולת שלנו למדל איזשהו הקשור בטוויחים אורכים, בהקשר לקונפלוציות הדבר העיקרי שדיברנו עליו היה שאנו מבטאים או מדברים איזשהו משקל יתר בין קלטים סטוקיים מתוק הבנה שהתלוויות אמורים ליצור משתנים מסבירים שהם יותר משמעותיים לעומת איברים רחוקים, לדוגמה בתמונה בדרך כלל נרצה להגיד שאיברים סטוקיים ננראים שייכים לאותו אובייקט לעומת איברים רחוקים מאוד שבדרך כלל יהיו שייכים לחלקים שונים בתמונה ולא הכרח קשור ביניהם.

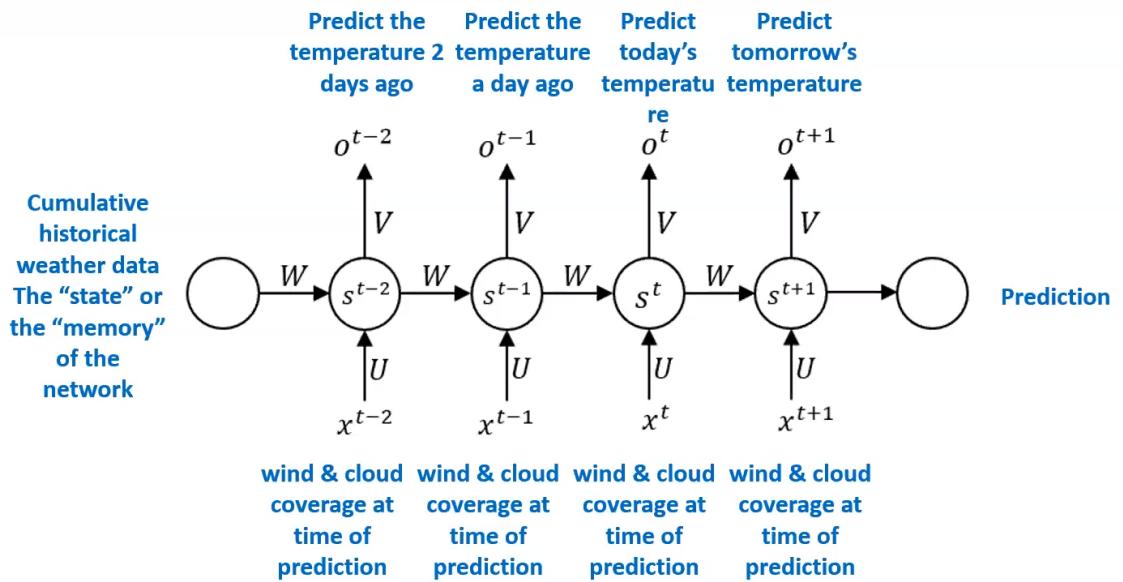
اذ איך נראה בעצם תהליך של RNN?



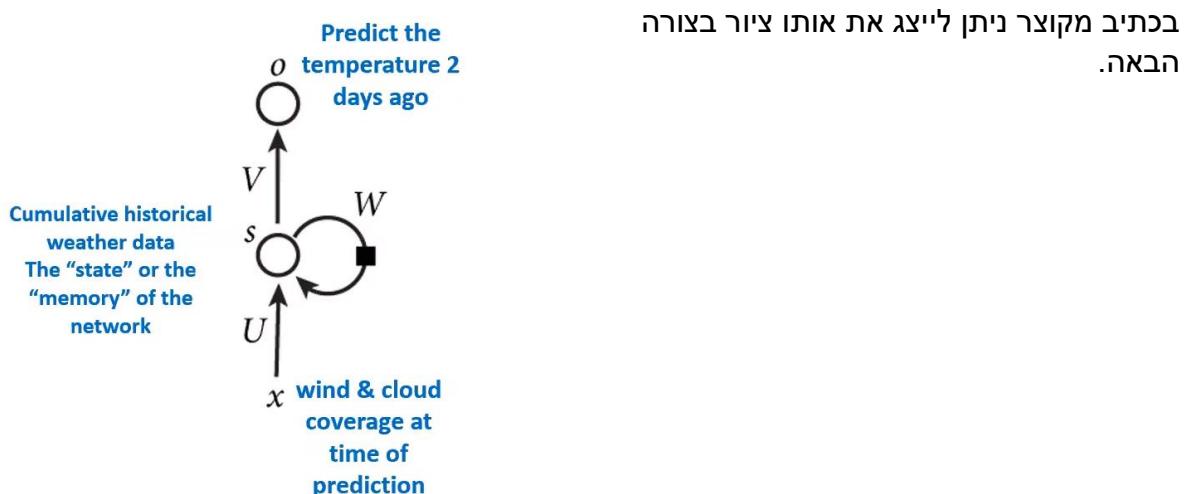
לאחר מכן נשימוש באותו מטריצות W , U ו- V כדי לשקל את state_{t-1} הקודם שקיבלנו ואת הקלט בנקודת הזמן הנוכחי וכן לייצר את הפלט לנקודת הזמן החדשה. וכך מתקיים עוד ועוד שלבים.



דוג' לשימוש שנוכל לעשות בעזרה רשות צאת'.



כלומר עבור כל נקודות זמן, בעזרת נתוני הרוח והעננות נרצה לחזות את הטמפרטורה באותו יום.

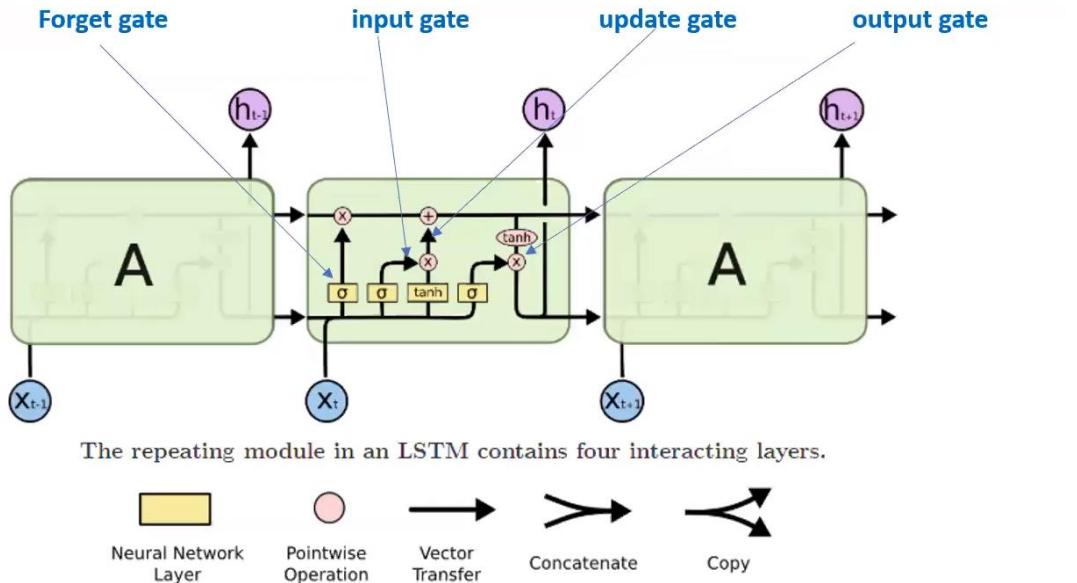


בעיה שאנו רואים בRNN:

ברשותות RNN הקלאו"ם, קשה מאוד לשמור את הסיגנל אחורה.
בדומה ל - ResNet אנחנו מנסים לחשב על-air להחזיק את הסיגnal למרחוקים ארוכים.

LSTMs basics

בעצם, כבר ב – 99 , יorgan שמיידובר, עלה במעבדה שלו על הרעיון לייצר כל מני שערים לוגיים שאומרים האם אנחנו צריכים לשמור את state הקיים, או שלא כדאי לשמור את state הקודם. כמו כן, האם ועד כמה אני צריך להתייחס לקלט הנוכחי, הרוי הקלט כולל outliers והוא יכול להשיב את כל הסדרה. גם כאן יש איזשהו שער לוגי שאומר האם ועד כמה להתחשב בקלט הנוכחי.



הבעיה שאנחנו מנסים להתמודד איתה זה שימור המידע לאורק נקודות שונות בסיגナル ועודין להיפטר מידע שלא רלוונטי, או מدلגים על נקודות שהן לא רלוונטיות בשביל לשמר את הסיגナル ואת הפלטים (כמו בResNet).

לטובות זה יש לנו ארבעה שערים לוגיים:

- **Forget gate:** בעצם אנחנו לוקחים את state הקיים ומכפילים באיזשהו שער לוגי, ורק אם הוא עבר את אותה נקודה לוגית (מעבר סף מסוים) אז רק אז נוסיף אותו לstate הנוכחי שקיבלנו מהשלב הקודם של הרשות.
- **Input gate:** מחליט אם מתיחסים לקלט הנוכחי או מתעלמים ממנו.
- **Update gate:** מחליט איך אנחנו מעדכנים את state שקיבלנו בנקודת הזמן הנוכחיית
- **Output gate:** מחליט מה אנחנו נוציא החוצה בתור פלט.

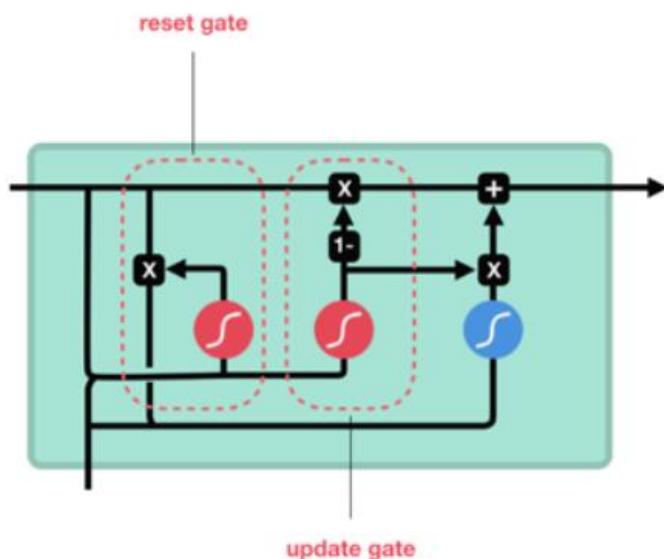
בצורה הזאת אנחנו בעצם מאפשרים להתייחס ל – sequences ארוכים יותר, רק נסיג ונגיד שגם אנחנו לא מצליחים ליצג sequences ארוכים מדי, כשתעלה למספר של נגיד 1000 או 2000 נקודות אז עדין תהיה לנו בעיה של שימור המידע.

GRU

ב-GRU ישנו רק שלושה gates, במקום ארבעה כמו ראיינו ב-LSTM, כך שיש תכולות של reset gate שזה איחוד של:

- Multivariate – קלט שיש בו יותר מערוץ אחד.
- לדוג' בערוץ בודד – להסתכל על ערכי מניה בודדת לאורך ההיסטוריה.
- כאן אנחנו שואלים את המודל – אנחנו לא רוצים לחזות את הנק' הבא, ככלומר אנחנו לא רוצים פרדיקציה בודדת, لكن רוצים פורנרטה forecast horizon? כמו בתחזית החדשנות, אנחנו רוצים לראות את התחזית לימים הקרובים, לדוג' בחלוקת לערים/חלקי שעות, זה עוד ערכאים.

טוב עברו משימות סיווג ולא עברו משימות גרסיה כי הוא לא מתיחס למרחק של הפרדיקציה.



הרצאה 6

בברצאה הקודמת ראיינו דוג' למשתמשים שצופים בסרטים ונותנים איזשהי ביקורת, ובמציאות שיטת אופטימיזציה של קולברינטינג פילטרינג אנחנו מגיעים לאיזשהו ייצוג חדש של משתמשים וייצוג חדש של סרטים בצדורה שיש איזשהי משמעות סמנטית שモובעת באמצעות אותו ייצוג חדש, ככלומר במקומות ליצג משתמש מסוים בודד, אנחנו יכולים להסתכל על היצוג הווקטוריו שלו בתור קומבינציה של מאפיינים של אותו משתמש ובאותה צורה אנחנו יכולים להסתכל על היצוג הווקטוריו של סרט כמו איזשהו שילוב של מאפיינים של אותו הסרט.

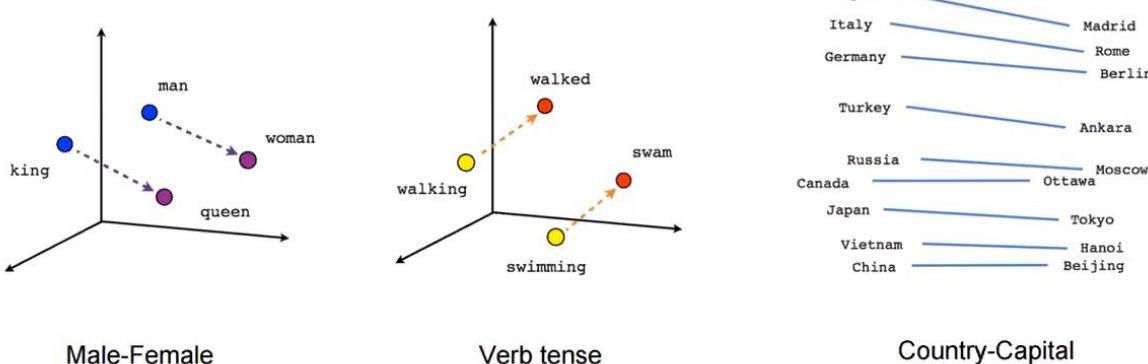
In the previous lesson:

user	movieId	rating	timestamp		user	movieId	rating	timestamp
73	1097	4	1255504951		73	113	1.96	1.14
561	924	3.5	177695223		561	132	0.87	1.01
157	260	3.5	191596691		157	100	0.99	1.01
358	1210	5	957481884		358	110	1.00	1.00
130	316	2	1138999234		130	116	4	120563146
580	1196	4	120563146		580	1196	4	120563146
544	2918	5	1435787004		544	113	0.95	0.99
123	1200	3	1462563045		123	1200	0.95	0.99
176	2571	4.5	1340714691		176	1200	1.00	0.95
481	4886	4.5	1340722207		481	1200	1.00	0.95
423	1270	3	1353700413		423	1270	0.90	0.95
607	364	3	1344791031		607	364	0.90	0.95
105	4963	3	1086699708		105	4963	0.90	0.95
165	500	3	1118842850		165	500	0.90	0.95
665	32	4	956233393		665	32	0.90	0.95
547	2997	5	974779039		547	2997	0.90	0.95
292	780	4.5	1140051052		292	780	0.90	0.95
134	231	2.5	1261245725		134	231	0.90	0.95

User representation					
14	0.19	0.63	0.31	0.44	0.51
29	0.25	0.83	0.71	0.96	0.59
72	0.30	0.44	0.19	0.00	0.72
211	0.02	0.72	0.69	0.35	0.25
212	0.60	0.87	0.76	0.36	0.04
293	0.73	0.70	0.44	0.47	0.29
310	0.23	0.81	0.36	0.47	0.12
379	0.68	0.90	0.20	0.92	0.74
451	0.81	0.41	0.81	0.15	0.17
467	0.70	0.61	0.90	0.89	0.24
508	0.50	0.27	0.73	0.44	0.83
546	0.16	0.21	0.75	0.48	0.98
563	0.91	0.75	0.75	0.24	0.06
579	0.55	0.58	0.68	0.93	0.66
623	0.94	0.25	0.46	0.16	0.30
movie representation					
27	0.71	0.81	0.74	0.04	0.04
49	0.92	0.55	0.86	0.44	0.80
57	0.68	0.28	0.53	0.16	0.94
72	0.83	0.88	0.33	0.41	0.24
79	0.60	0.50	0.81	0.73	0.53
89	0.18	0.31	0.68	0.39	0.09
92	0.26	0.08	0.92	0.29	0.74
99	0.91	0.47	0.61	0.94	0.13
143	0.99	0.94	0.46	0.12	0.39
179	0.52	0.70	0.64	0.67	0.44
180	0.91	0.11	0.24	0.54	0.81
197	0.53	0.87	0.25	0.57	0.80
402	0.23	0.20	0.83	0.52	0.23

את אותו רעיון ניתן לעשות גם עבור מילימ, אם נחשב על זה, משתמשים וסרטים יש לנו בסך הכל התנהגוויותיחסית דומות.

במילים נרצה ליצג את המרחב הגדול הזה (אם באנגלית לשם הדוג' יש כשתים מיליון מילימטרים רוחב) בצורה שהיא הרבה יותר שימושית עבורנו, נראה גם שהיצוג מחדש יכול ליצר עבורנו התנגדות מענינית בצורה של הייצוג שנוכל ממש לעשות פעולות ארכיטומיות בין מילימטרים (לדוג' king ו-queen) לבין מילימטרים (man ו-woman).



למה שנרצה לעבוד עם עוד ייצוג של מילים?

- לא רוצים לעבוד עם דברים מאוד ספרטיים, בדרך כלל מעmis מאוד על הזיכרונות.
- יכולת ההבעה שלנו במשמעות סמנטית ולא משמעות סינטקטית. לא תמיד נרצה ל揖ג במלחינים שהתבלבלנו אויל בטעות בהקלדה של אות ואז לעשות השוואה, אלא בין מילים שיש קרבה סמנטית שהמשמעות שליהן דומה ולוו דוקא המרחק של כל אות לאות מקבילה לה.
- יש הרבה שימושים של word embeddings שאנחנו בדומה לזה שאנחנו יכולים לטעון משקولات מוכנים, בהתאם לארQUITקטורה שאנחנו מאמנים אנחנו יכולים לטעון גם embedding אם לתוך גרפ או לא חייב לגרף ואנחנו נראה בהערכתה הزادת שלושה מימושים לדבר זה.

Word2Vec -

GloVe (Global Vectors for semantic text representation) -

FastText -

Some background – Bag of words & TFIDF

- Counting methods – we use counts of occurrences to express the meaning of a word or a phrase

Document 1: "The cat sat on the hat"

Document 2: "The dog ate the cat and the hat"

	the	cat	sat	on	hat	dog	ate	and
Document 1	2	1	1	1	1	0	0	0
Document 2	3	1	0	0	1	1	1	1

- TFIDF

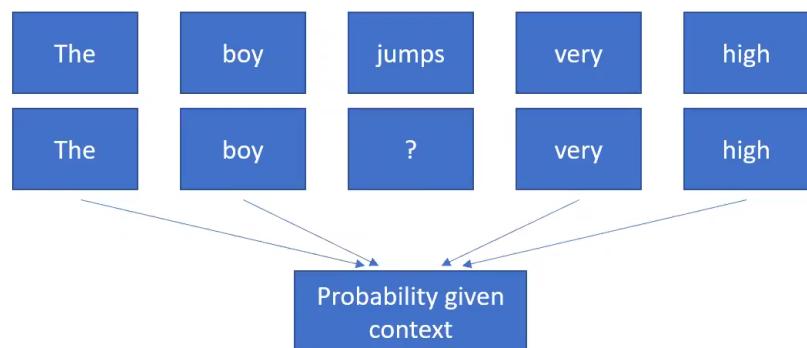
TF = (Number of times term t appears in a document)/(Number of terms in the document)

IDF = $\log(N/n)$, where, N is the total number of documents and n is the number of documents a term t has appeared in.

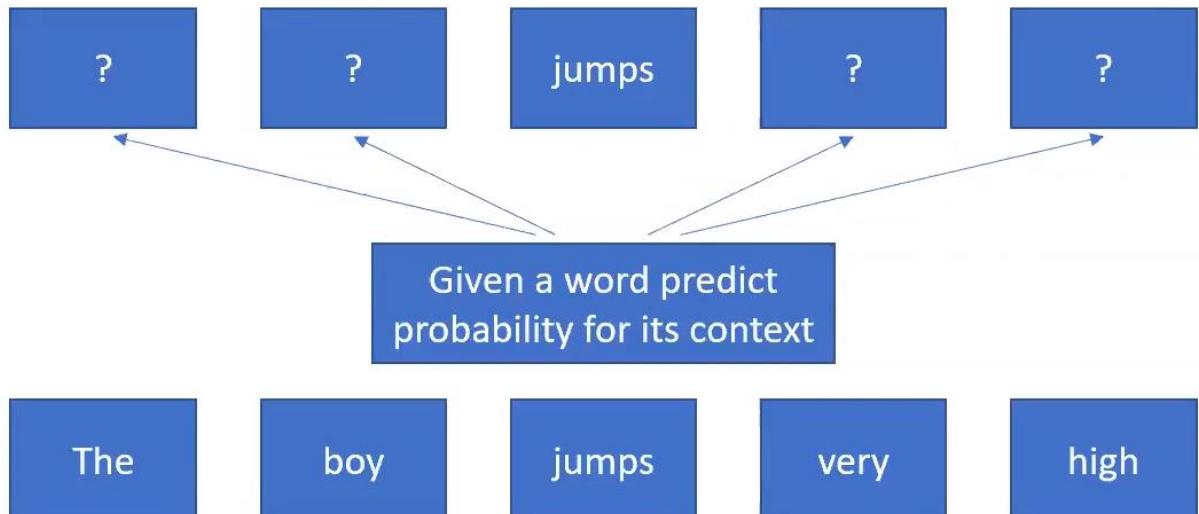
TF-IDF(t, document) = TF(t, document) * IDF(t)

Word2Vec

הרעין הבסיסי מציג שתי שיטות אלטרנטיביות, השיטה הראשונה (CBOW) מנסה לחזות מה ההסתברות למילה להיות מילה מסוימת בהינתן השכנים של אותה המילה.



השיטה השנייה (Skip-Gram) אומרת הפוך, תגיד לי מה המילה שיש לך במקום מסוים, ואני אנסה לחשות את הסבירות לכל אחת מהמילות האחרות להיות שכנה.



Glove

שיטה נוספת לקבול word embedding נקראה Glove או קיצור של global vectors ומה שהיא רואים זה איזשהו ניסיון, להציג word embedding בתור יציג סטטיסטי שmboso על הסתברות המוטנית של מילה מסוימת בהינתן מילה אחרת שקדמה לה.

לדוגמא, מה ההסתברות שהמילה הבאה תהיה המילה `ice` בהינתן המילה `solid`? ואז אנחנו מקבלים וקטור הסתברויות בהתאם למילימ האפשריות.

- Aim – express semantic meaning using word vectors
- Means – usage of global count statistics
- Learns its representation using the co-occurrence matrix
- Loss is weighted according to word frequency

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k ice)/P(k steam)$	8.9	8.5×10^{-2}	1.36	0.96

FastText

הרעיון המרכזי דומה קצת לWord2Vec, בשונה אבל היא משתמשת באיזה תובנה Bair מילים בניוית, חלק מילה מסוימים יכולים לתת לנו מידע לגבי שאר המילה (לדוגמא מילים שנגמורות ב-ing) וככה למדוד קשר בין מילים שלא בהכרח קרובות במשמעות שלهن, אבל יכול להיות שיש איזשה' קרבה סמנטית.

היתרון הגדול בשימוש בשיטה זאת היא שניתן להכליל טוב יותר, כלומר נוכל למדוד מילים וביטויים בעלי שכיחות נמוכה יותר, וכך מילים או ביטויים שהשכיחות קרובה ל-1 יהיה קשה ללמידה, אבל ככה נוכל לייצג אותם בצורה די טובה.

כמובן שהוא דבר לא מקבלים בחינם, והחיסרון המשמעותי הוא זמן אימון המודול.

Basic concepts:

- Using a word2vec-like model
- Combining results for word level and character level modeling

Apple -> <ap”, “app”, “appl”, “apple”, “apple>”, “ppl”, “pple”, “pple>”, “ple”, “ple>”, “le>

Advantages:

- Better results in general
- Better results for rare labels
- Ability to deal (to some extent) with OOV terms

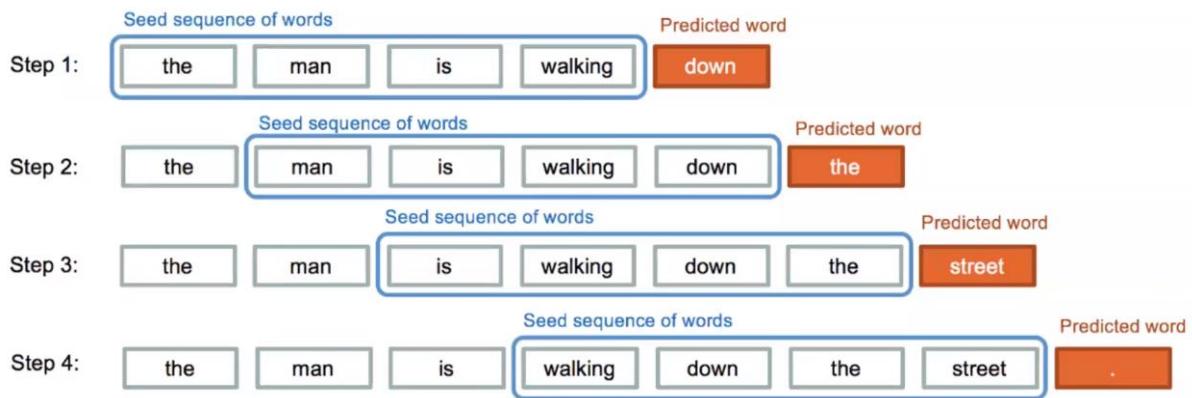
Main disadvantage:

- Longer to train

Text generation using LSTMs

Text generation מtabסס על הרעיון הבא, אנחנו יכולים להסתכל על רצף של מילים, או שאנו יכולים להסתכל על רצף של אותיות ומה שאנו יכולים לומר זה בהינתן איזהו קלט מה תהיה המילה או האות הבאה.

אפשר להסתכל על זה כבונית קלסיפיקציה אם אנחנו מסתכלים על מספור, או כבונית רגסיבית אם אנחנו מנסים לחזות מה תהיה המילה הבאה בהקשר סמנטי, אנחנו יכולים לעשות גרסיה לכל אחד מהמרכיבים של כל אחד מהיצוג הווקטורי של המילה הבאה, ולהסתכל מה המילה הקורבה ביותר באותו מקום בהיפר ספרס שהגענו אליו.



בעיה אפשרית פה שזה אנחנו יכולים להכנס לпот אינסופי של חזרתיות, נניח שבמוקם שבשלב 4 אנחנו נחזה את המילה the אנחנו יכולים לחזור למצב שאנוחנו בהמשך חוזים את המילים, is, man, walking וכה עצם לחזור על עצמנו שוב ושוב.

Fully Convolutional Nets, Image Segmentation, Localization, Detection

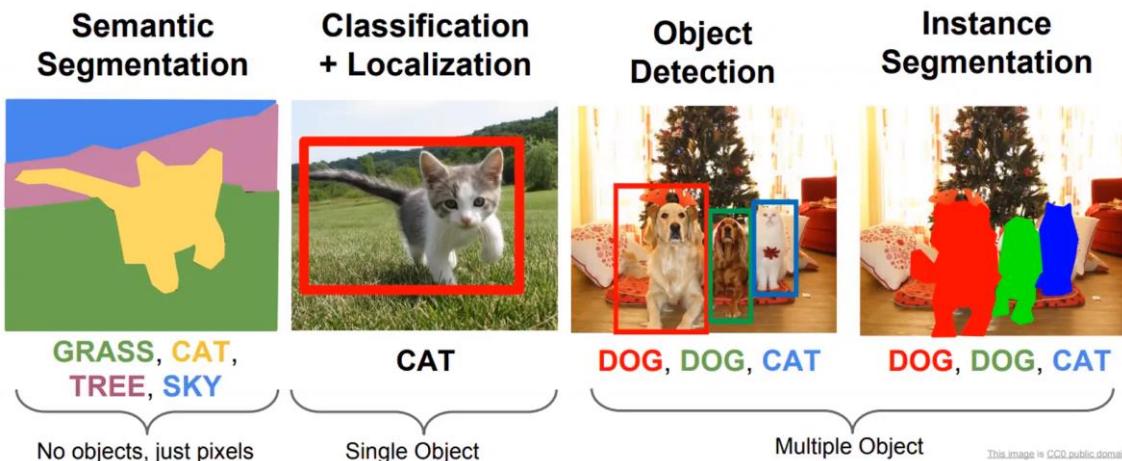
עד עכשוו שדיברנו על משלימות של computer vision דיברנו על שני סוגים עיקריים של משלימות.

- סוג ראשון שנקרא Classifications
- סוג שני שנקרא regression

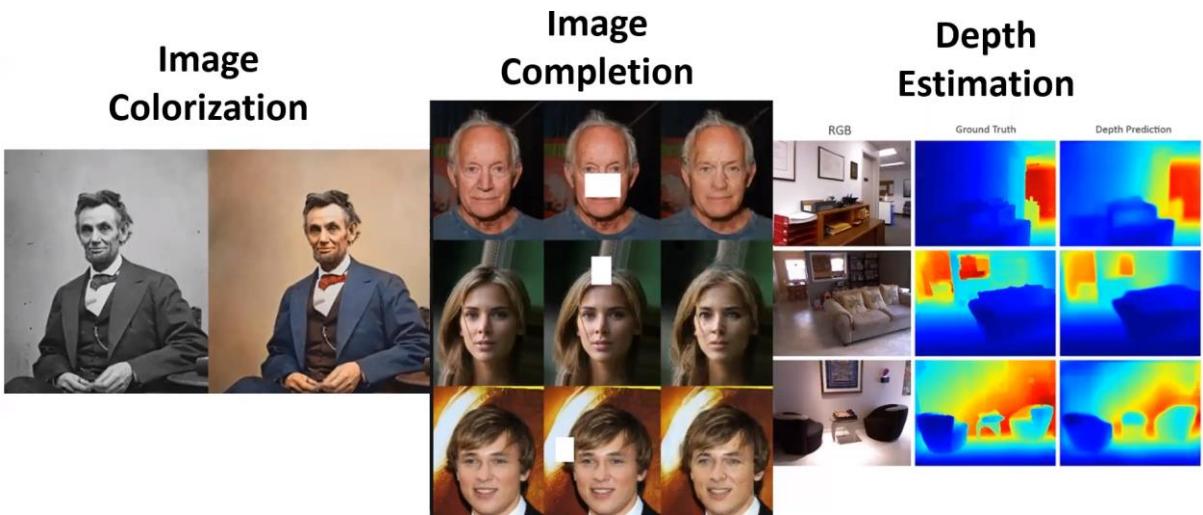
בהרצאה הצעתי נרצה לצאת מתוך אותו "קיבעון" מחשבתי, שבעיות שאנו פותרים באמצעות deep learning משיכות רק לאוון שתי קטגוריות ונתחיל ליצור לעצמו יכולות, להגדיר בעיות בצורה שונה ולהשתמש בכלים שלנו מתוך השתי משפחות האלה כדי לפתרו סוגים חדשים של בעיות.

סוגים כאלה של בעיות יהיו:

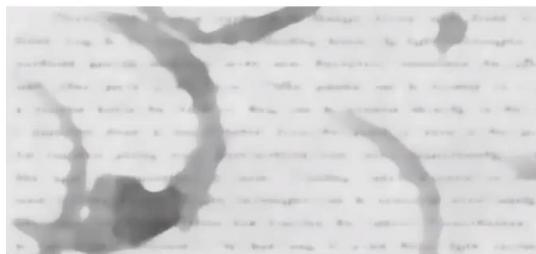
- Semantic Segmentation, שהרעיון כאן בהינתן איזשה תמונה, נרצה לבצע סגמנטציה בrama של כל פיקסל בודד. כמוור אמר קודם ביצענו סיוג לתמונה כולה, אך כאן עבור כל פיקסל נרצה לומר לאיזה קטgorיה הוא שייך.
- Classification + Localization, מנהיים שיש אובייקט אחדמשמעותי בתמונה, או אובייקט אחד שמשמעותו אונטנו בתמונה ומ Chapman את המיקום שלו בתוך התמונה, בהינתן שakan מצאנו אותו נרצה בזוזה לסייע לנו לקטgorיה הרלוונטית אליו.
- Object Detection, בעצם שיש לנו ריבוי אובייקטים ובפוטנציה גם ריבוי קטgorיות. במקרה זה יש לנו ניסיון לאתגר אזורים שונים בתמונה שיש בהם אובייקטים שונים.
- Semantic Segmentation, השילוב של Object detection ביחד עם Semantic Segmentation מביא אותנו לבעה האחורה, שכאן בדומה ל Semantic Segmentation אנחנו נרצה שהיא לנו סיוג ברמת פיקסל אבל גם שיש לנו שני מופעים מאותו קטgorיה, לא נרצה שהם יסווגו באותו אופן, אנו נרצה שהם יופיעו כסיוג נפרד של אותה קטgorיה.



Other Computer Vision Tasks

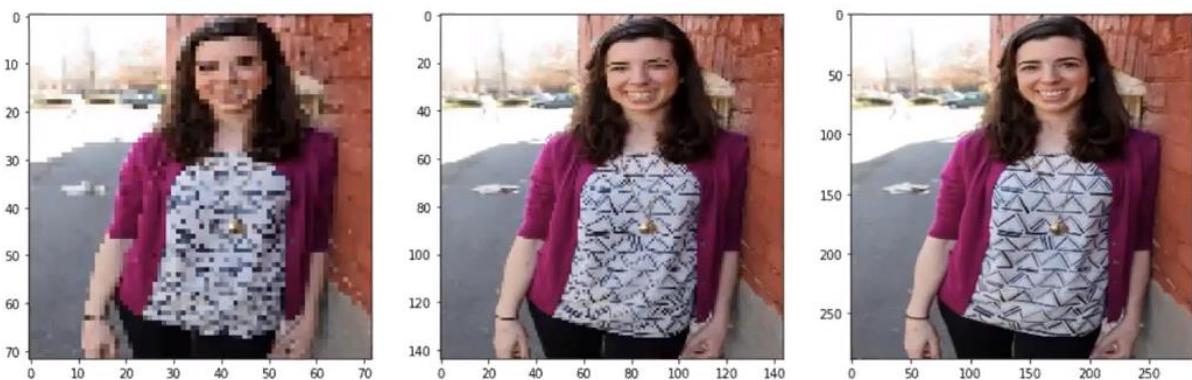


– לקחת תמונה עם רעש (לדוגמא מכתב שיש עליו כתמי קפה, ולשזר את הטקסט המקורי, או את אותו המכתב ללא הכתמים)



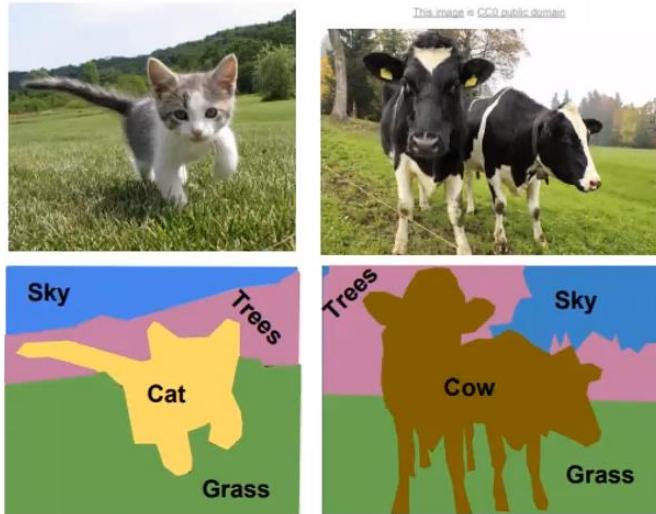
There exist several methods to design forms with fields to fill in. Some methods may be surrounded by bounding boxes, by light rectangles or other shapes. Some methods specify where to write and, therefore, minimize the effort required to align the text with other parts of the form. These guides can be located on a separate sheet or on the same sheet as the form. A separate sheet is much better from the point of view of the quality of the text, but requires giving more instructions and, more importantly, rescaling the text. This type of acquisition is used. Guiding rulers printed on the page are used for this reason. Light rectangles can be removed more easily whenever the handwritten text touches the rulers. Nevertheless, they must be taken into account: The best way to print these light rectangles is to use a high-resolution printer and to print them on a separate sheet.

– לקחת תמונה ברזולוציה נמוכה ולהפוך להRAW לRAW��olution



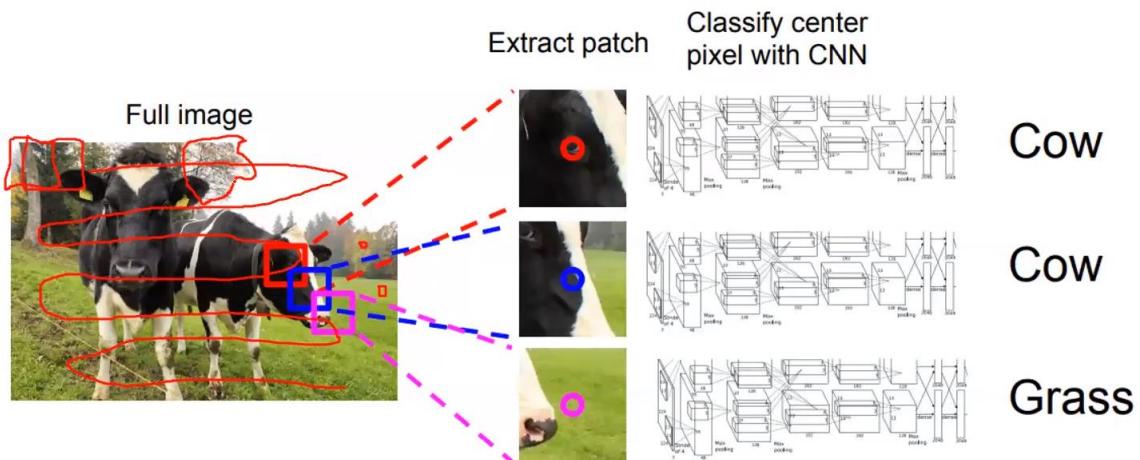
Semantic Segmentation

בהינתן איזהו תמונה מקור, אנחנו נרצה לסואג כל אחד מהפיקסלים מהתמונה המקורית לאחת מ-
הקטגוריות האפשריות.



כתנאי בסיסי אנחנו נרצה שהתיאוג שלנו
יהה ברמת גורנליות הרבה יותר גבוהה,
לעומת תיאוג בודד עבור כל התמונה, אנחנו
מדוברים על תיאוג של כל פיקסל ופיקסל למה
הוא שיר.

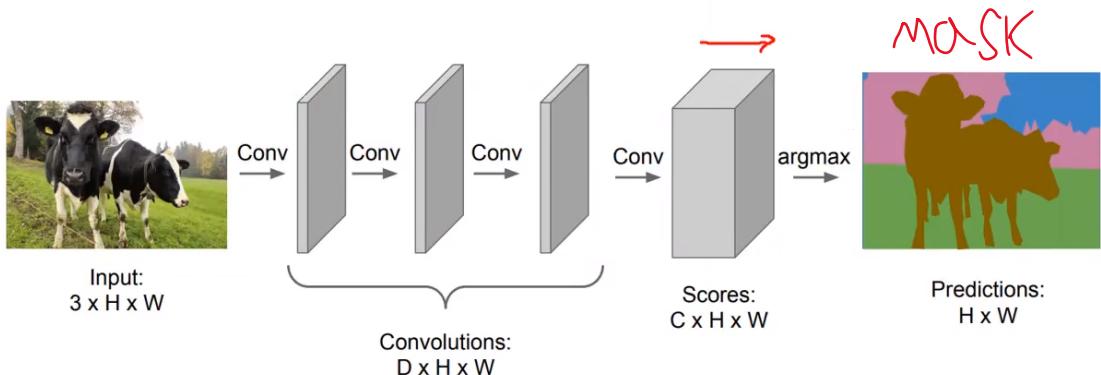
Naïve approach – לעבור עם גודל מסגרת על התמונה, ובכל חלק בעזרת רשת CNN לנסוט
להגיד לאיזה חלק אותה בדיקה שייכת, גישה זאת בעייתית עם מסגרת גדולה מדי ויכולת לתת לנו
רק תוצאה אחת בודדת (לדוג' פרה), במסגרת קטנה מדי יכולה להיות בעיתית לזהות מצבים בין
אובייקטים שונים (לדוג' דשא/עצים) ומשהו בגודל ביניהם יכול עדין למנוע מאיינו לקבל את מה
שנגיד "חלקות" בתמונה ולזהות קצותוויות שונות לאובייקטים שונים.



Another idea – Fully Convolutional Network (FCN)

דרך אחרת לבצע את אותה משימה יכולה להתבטא בשימוש ברשתות שנקראות **Fully Convolutional Network**, הרעיון המרכזיפה זה שאין לנו פעולה של flattening, כל הפעולות שאנו עושים אלה פועלות קובולוציה ולק התוצאה שנקבל יכולה להיות גם היא עם ממדים מרחבים, כמובן אם נבצע מספר רב של קובולציות ובשלב האחרון יש לנו מספיק עוזרים ככמויות הקטגוריות שיש לנו בסופו של דבר איז אוננו יכולים להניח שהפעולה של argmax מעלה כמות העוזרים הזו יכול להביא אותנו לאותו מרחב פתרון שאנונו מחשפים בmask.

(mask – זה בדיקת התיאוג שלנו, אוננו רצים תיאוג ברמת כל פיקסל מהתמונה המקורית ולאיזה קטgorיה הוא משתייך)



דבר נוסף מעניין שניtin לשים לבזה שבתמונה בצד שמאל למעלה יש הבלחה קטנה של שמיים, שנראה לא היינו מצלחים לטייג בצורה צאתית בשיטה הקודמת שהצענו לעומת כן שהרשות כן מצליחה.

שאלה: איך הגענו מ $M \times H \times 3$ ל $W \times H$ ולא קטנוו בגודל של הממדים המרחביים?

תשובה: ריפדנו כMOVON (padding).

Fully Convolutional with Down + Up sampling

בשלב הראשון נרצה להקטין את הרוחולציה באמצעות MaxPooling אבל מגיעים לממד عمוק גדול יותר ולאחר מכן לעשות את הפעולה ההופוכה UpSampling, כמובן גגדיל את הרוחולציה אבל נקטין את ממד העומק.

מה האתגר הגדול שיש עם UpSampling? – בעצם הפעולה של MaxPooling היא צמצום המידע שהוא חזקיקים, וזה די קל ואפיון ניתן לעשות את זה בצורה חכמה, לעומת זאת יש צורך לשנות פעולה של UpSampling המורכבות העיקרית היא כיצד ליצור מידע במקום שלא נמצא בראשותינו מלכתחילה. וכך אנו לא רק מתבקשים ליצור מידע אלא ליצור מידע בהתאם לממד הצבע שנמצאים באותו אזור, אלא גם במובן הקונספטואלי של מה שנמצא שם.

UnPooling operations

כיצד נבצע את אותה אופרציה של UpSampling

יש לנו מספר שיטות אפשריות:

Nearest Neighbor

1	2
3	4



1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Input: 2 x 2

Output: 4 x 4

- Nearest Neighbor -

משמע אני רוצה להכפיל את הממדים מ- 2×2 ל- 4×4 אך אני מכפיל כל אחד מהפיקסלים פי 4 ומקבל פלט שמתאים בממדים ומשכפל את אותם ערכים שכבר יש לנו.

“Bed of Nails”

1	2
3	4



1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Input: 2 x 2

Output: 4 x 4

- Bed of Nails -

נחשב שככל אחת מהנקודות נרפס באפסים, ונשיר את הנקודה הנוכחית לפינה השמאלית העליונה בדוג' הספציפית הזאת.

Max Unpooling

הרעין הוא שאנו יכולים להגיד Maipe ביצענו את pooling Max
כלומר שאם נרצה להגדיל את הפלט שלנו אז בשונה מ-*Bed of Nails* בהתאם לשיטת pooling Max אנו נדע לשים את הערך הקיים שלנו.

Max Pooling

Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4

Output: 2 x 2

Max Unpooling

Use positions from pooling layer

1	2
3	4

Rest of the network

0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

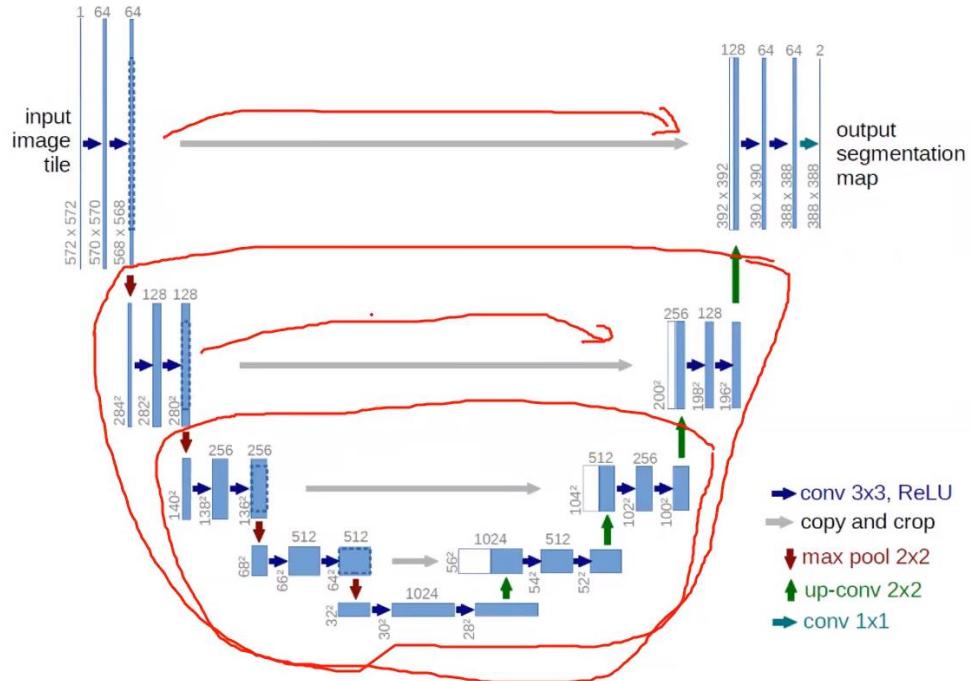
Input: 2 x 2

Output: 4 x 4

Semantic segmentation – UNet

הרשת היכי נפוצה בעולם היזאת של Semantic segmentation היא רשת שנקראת UNet ורשת זאת היא מקרה פרטי של שמלב גם קונספיטים של autoencoder (רק נזכיר שהה עצם רצון לחזות את הקלט המקורי באמצעות איזה מרחב מצומצם של אותו קלט מקורי, כלומר מבצעת דחיסה של אותו קלט ולאחר מכן פענוו של אותו קלט דחוס בשביל לשחרר אותו מידע דחוס).

از הרעיון של ארכיטקטורת UNet, הוא שאנו מנסים להשתמש ב skip connections (כמו במבנה שמציר autoencoder , שההבדל היחיד שהוא מוספים skip connections (כמו ResNet).



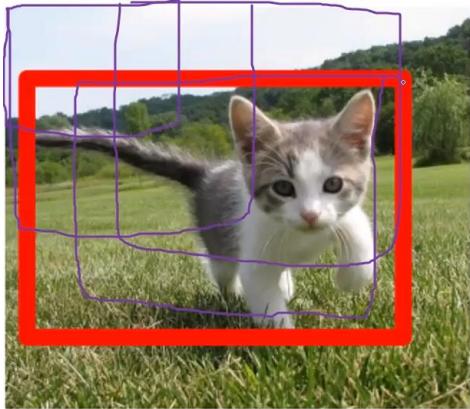
בכל שלב אם אנחנו לא נוסיף מידע רלוונטי אנחנו פשוט נעביר את המידע המקורי כמו שהוא.

Object Localization

מספר אפשרויות כיצד לבצע מטלה מסווג זה:

Sliding window -

לבצע תהליך של העברת חלונות וכל פעם להחזיר הסתברות לגבי האובייקט הפנימי, ובשאייפה שנצאליה להחזיר חלון עם "יצוג cocci" גבוה לגבי הימצאות של האובייקט בתמונה.



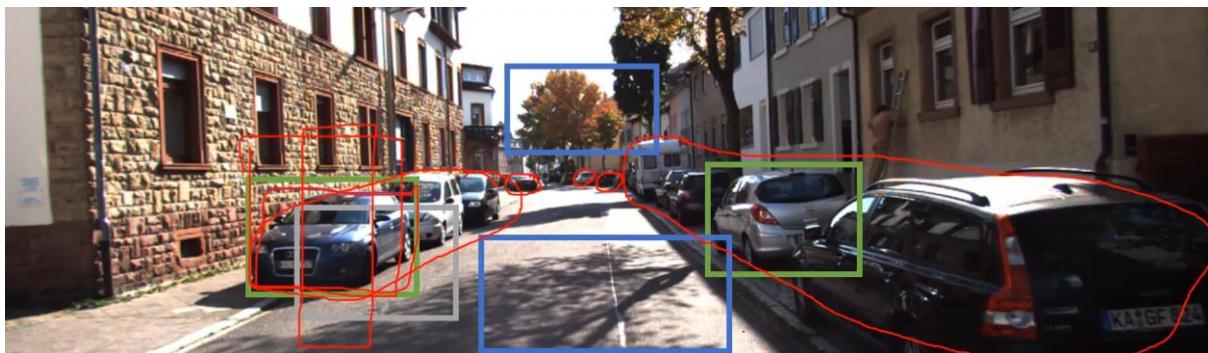
Bounding box - נבצע בעיית

גרסיה ונחפש שתי נקודות, שמאל עליונה וימין תחתונה וככה נתחום את האובייקט שלנו.

אתגרים קיימים

Detection is harder than classification:

- Localization errors
- Huge class imbalance
- Variation in scale and aspect ratios
- Tricky occlusions
- No clue for the number of objects

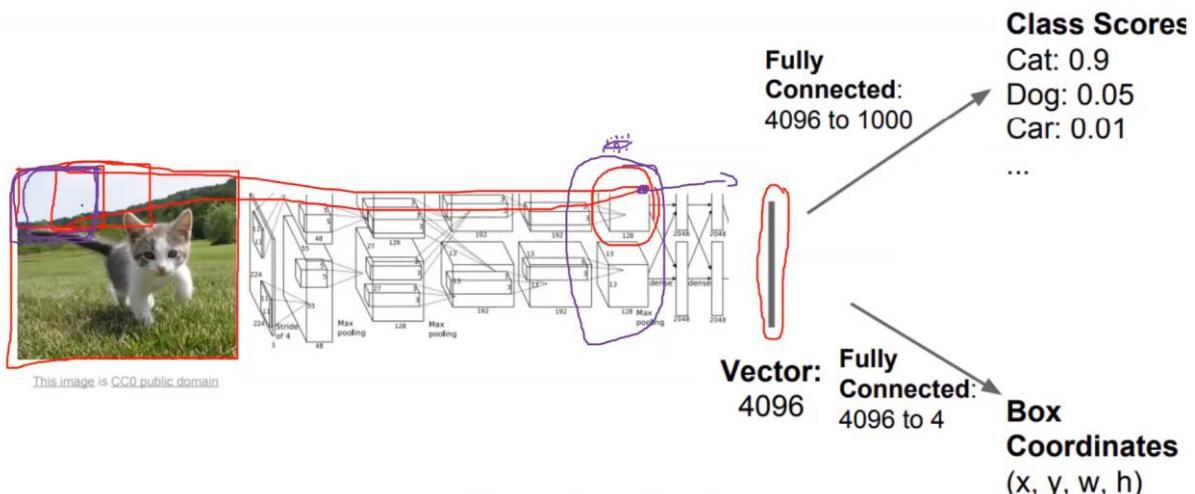


Sliding-window



הבעיה העיקרית פה זה גמ שזה תהליך מאד ארוך
שיכול לחתת הרבה מאד זמן, ובנוסף יכול להיות מצב
שהחלון שליל לא מותאם לגודל האובייקט ואז אני פשוט
אפסוף אותו, אול אני משתמש.stride גדול מדי אבל
יכול להיות שגם עם stride מותאים אני פשוט לא אצליח
לאסוף מספיק פיקסלים על מנת לקבל הינה מספיק
טובה לגבי האובייקט שני רצח לדעת מה הוא.

מה שאנו נרצה לומר בסופו של דבר עבר כל תמונה בקלט מה הם האזוריים שיש לנו פרדיקציה מעניינת, פעולה זו אנו מסוגלים לעשות על כל התמונה יחד ובמקום לחשב את הקונבולוציות על כל חלק בנפרד, נוכל לחשב פעמי אחת את כל השכבות הראשונות על כל התמונה, לקבל פיצ'ר וקטור שמעניין אותנו על כל התמונה, ובנוסף יש לנו את היצוג של כל חלק בתמונה, אז נוכל למפות מוקודה מסוימת לרדיוסטיב פילד, שהזין את אותה נקודה ואם עבור אותה נקודה מסוימת נקבל פרדיקציה של קלואס מסוים אז נוכל לשער את אותו אזור לדיטקשיון. לא נctrיך להריץ באופן נפרד את כל השאר הנקודות דרך הרשת, נוכל להשתמש ביצוג שקיבלנו בשלב האחרון בשביל ליצר לעצמנו פרדיקציות של האזוריים שימושיים מבחינת הזהה.

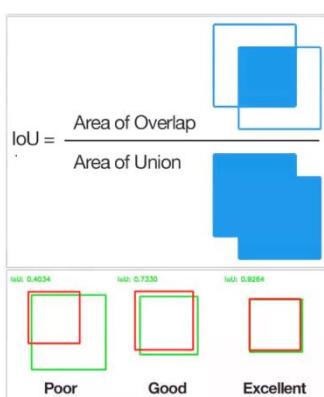


דברים נוספים שכדאי להזכיר:

- Region proposal: pick a subset of prospective regions and score them with a binary classifier
- Bounding box regression: predict the coordinates of the boxes as real-valued variables

IoU – Intersect Over Union

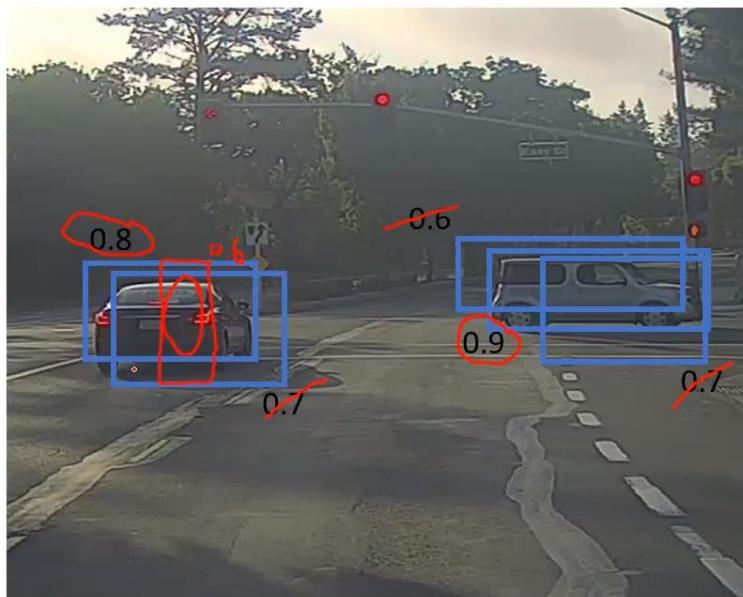
פונקציה זו ממחשבת את היחס בין intersection over lapping region לבין התיאוג והפרדיקציה.



- הזכרנו שמתוחת ל 0.5 זאת חפיפה לא טובה.
- מעל 0.7-0.85 מוגדר חפיפה שהיא די טובה
- ומעלה 0.85 ופרט 0.9 זה נחשב טוב מאוד.

Non – Max Suppression (NMS)

אלגוריתם מאד מוכר בעולם היזהוי שאומר את הדבר הבא - אנחנו בוחרים קודם כל את הפרדיקציה עם ההסתברות הגבוהה ביותר מתוך כלל היזהויים שיש לנו, עבור אותה פרדיקציה אנחנו בוחנים מה כלל היזהויים הנוספים שישים לאותה קטגוריה והם בחפיפה שמעל סף מסוים (במקרה שלנו מעל 50%) ובאזור זה את אנחנו פסלים את כל אותם תיוגים שהם בחפיפה ושיכים לאותו class.



בתמונה הבאה נתנו דוגמה שאת המכונית הימנית אנחנו מוצאים לפי האלגוריתם את ההסתברות המקסימלית (0.9) ופסלים את שאר ההסתברויות שנמצאות בחפיפה, במכונית השנייה הוספנו את האדם שיושב בתוך הרכב עם הסתברות של 0.6 ואז עבור אותה מכונית עם הסתברות 0.8 אנחנו פסלים את התיוגים שנמצאים בחפיפה ומתייחסים כרכב אבל את התיוג של האדם אנחנו לא פסלים.

Very popular in detection algorithms

We want to make sure each object is detected only once.

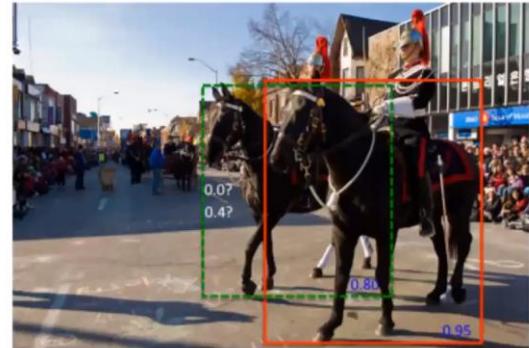
- *Input: set of detections ($\{B_i, P_i\}$)*
- *Sort in the decreasing order of P_i*
- *For $i = 1$ to N*
 - *Pick the bounding box i*
 - *Remove all boxes with $IoU > 50\%$ with box i*

A problem arises when multiple objects overlap significantly

Solution – instead of omitting each overlapping object reduce the detection probability relatively to the overlap

NMS

$$s_i = \begin{cases} s_i, & \text{iou}(\mathcal{M}, b_i) < N_t \\ 0, & \text{iou}(\mathcal{M}, b_i) \geq N_t \end{cases},$$



Soft NMS with linear function of the overlap

$$s_i = \begin{cases} s_i, & \text{iou}(\mathcal{M}, b_i) < N_t \\ s_i(1 - \text{iou}(\mathcal{M}, b_i)), & \text{iou}(\mathcal{M}, b_i) \geq N_t \end{cases},$$

Soft NMS with Gaussian function of the overlap

$$s_i = s_i e^{-\frac{\text{iou}(\mathcal{M}, b_i)^2}{\sigma}}, \forall b_i \notin \mathcal{D}$$

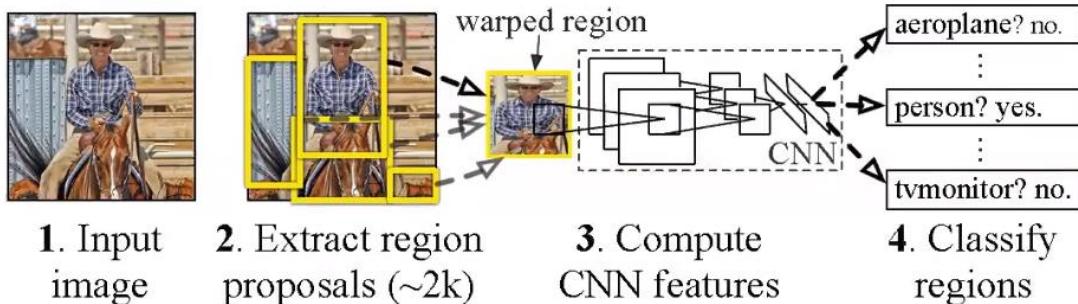
Object Detection Using ImageNet CNNs

הראשת הראשונה שהציגה פתרון לבנייה של לוקליזציה ודיטקשיין של אובייקטים בתמונות הייתה רשת שנראה RCNN (Regions with CNN features).

שלבי הפתרון:

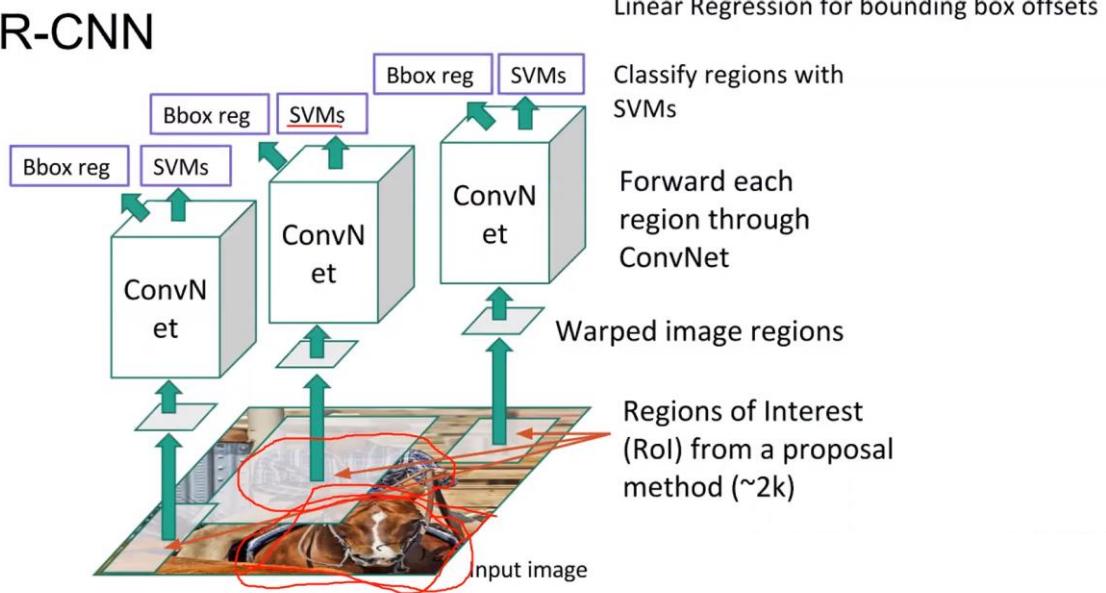
- נזכיר קודם כל Selective Search כדי להציג מה הם אזורים שיש בהם טקסטורה דומה או צבע דומה או דחיסות של פיקסלים בצבע דומה מסוימת ועל פי אותן מאפיינים זהים אנו נציג אזורים שהיו אמורים עניין אפשריים.
- נעביר כל אחד מהאזורים המוצעים לרשת CNN ונוציא את הפיצ'רים שלה, בעצם נקבל את הפיצ'ר וקטור מהרשת SVM בשכבה האחורונה של אותה רשת
- נבצע התאמה/הידוק של הקודינטות שמייצגות עבורנו את ה-bounding.

R-CNN: Regions with CNN features



הבנייה העיקרית בתחום זה הוא גם יהיה אורך ו גם קשה חישובית.

R-CNN



از כמו שכבר אמרנו הבעה העיקרית פה היא איטיות, הצורך האדיר בזיכרון ויכולות חישוב והאופי סידרתי מצד אחד והעבודה שאין לנו loss function אחד שמתכלה את כל הדברים, כל שלב עומד בפני עצמו.

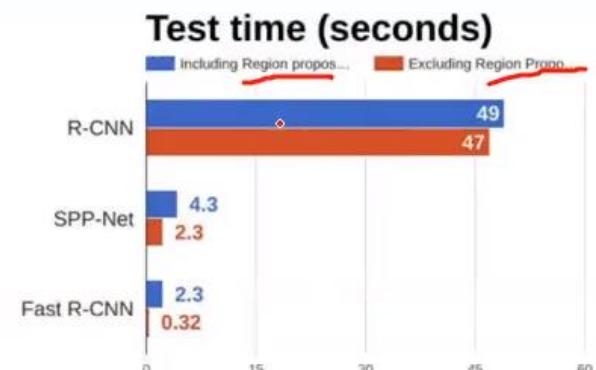
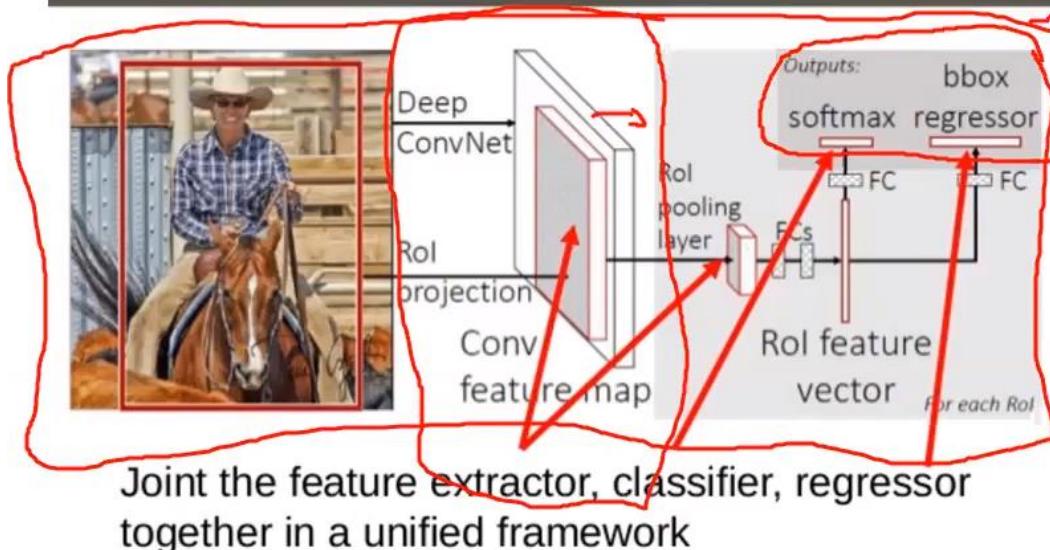
Fast R-CNN

השינוי המשמעותי הוא תכלול כל המשימות לכדי רשות אחת, אז יש לנו שכבה חדשה שנקראת Rol Pooling והרעיון שבහינתן הפיצ'ר וקטור שאנו מקבלים אחרי הרצת של רשות על פni התמונה בקילט, אנחנו מקבלים מתוך הפיצ'ר וקטור הצעה של איזורי עניין, מה יוגדר איזור עניין? בעצם בכל איזור אנחנו נחפש אותם פיקסלים שבספייס מאפ הם בעלי הערך הגבוה ביותר בטור הצעה לאיזור עניין. יש לנו שכבה שזה תפקידה.

דבר נוסף זה Multi-task loss, הטעיה שהוצגה קודם loss זה שהו loss בכל שלב משל עצמו, ולא היה שום דבר שתיכל אותו, ובעצם אנחנו יכולים לראות שעכשו כל התהילה מובא חלק אינטגרלי אחד.

השלב של יצירת איזורי עניין הפרק עצמו להיות הצואר בקבוק בראשת הزادתי, אבל מצד שני ניתן לראות שכך נוצר שיפור גם בזמן האימון מ-84 דקות לפחות ל-9 דקות ורובה יותר בשלב הטעון, שאנו רואים ירידת 49 דקות לתהילה כולה ל-2.3 דקות.

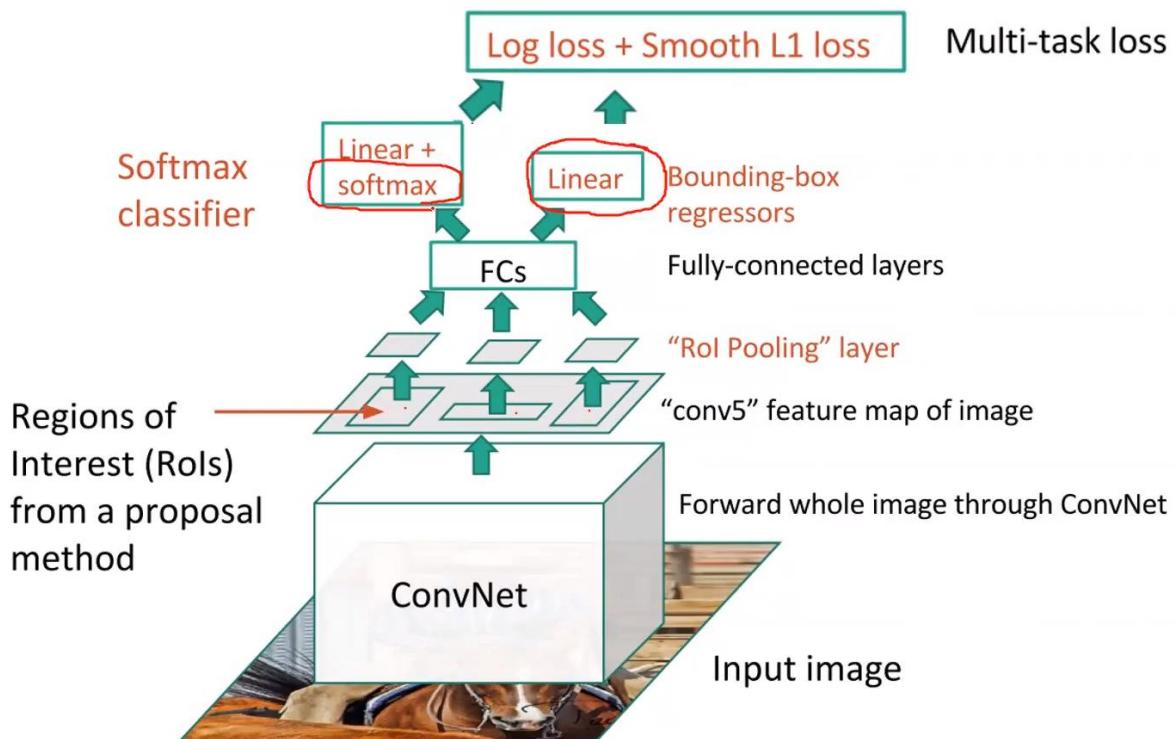
Fast R-CNN: Joint Training Framework



- To avoid recomputing convolution on regions, compute CNN features for entire image in one pass (instead of 2,000 times)

- As before, use *Selective Search* to propose regions

- Introduce a *Roi Pooling* layer
- Multi-task loss - output includes softmax for object detection + regressor bbox, trained together
- Now region proposals became the bottleneck – *Selective Search* is a slow algorithm

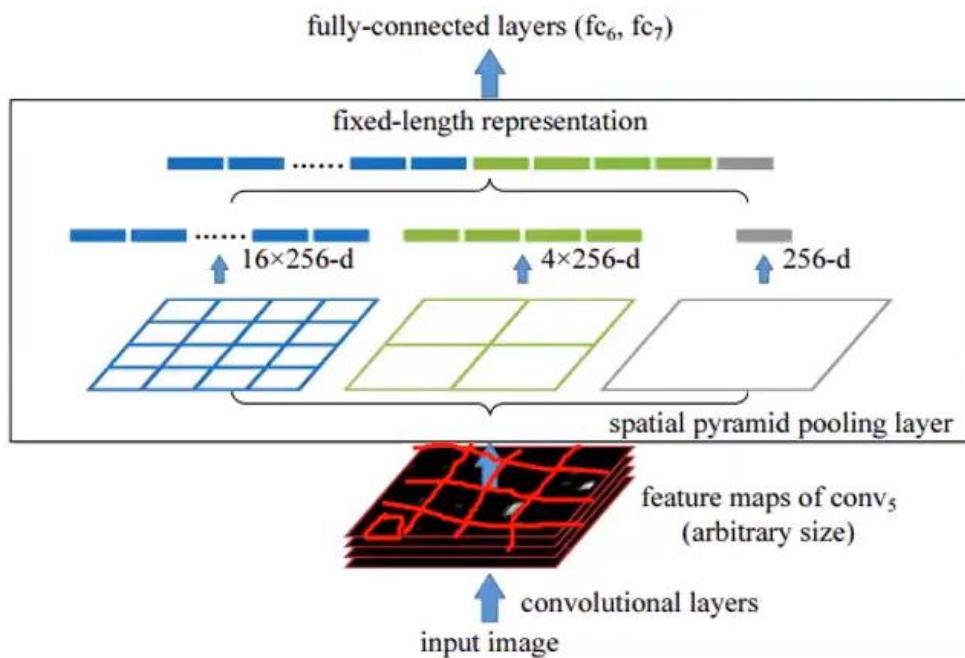


RoI Pool Layer

נמיר כל אזור עניין לתוך מפת פיצרים בגודל מסוים, כל ROI מוגדר באמצעות ארבעה משתנים, שמייצגים את הנקודה השמאלית העליונה וגובהו ורוחב.

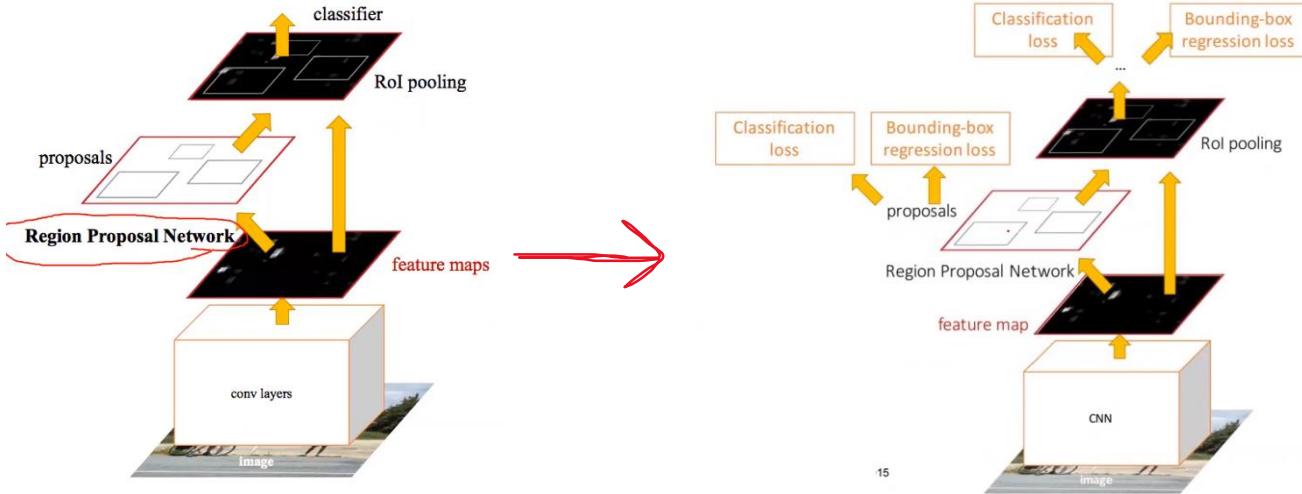
פועל על ידי חלוקת חלון ROI $w \times h$ לרשת $W \times H$ של חלונות משנה בגודל משוער $W/w \times H/h$ ולאחר מכן איחוד מקסימלי של הערכים בכל חלון משנה לתוך תא רשת פלט המתאים.

- אנחנו מחשבים את הפיצ'ר מאפ' עם אחת
- אין חישובים כפולים
- פחתת תלותי על מספר האזורי המוצעים
- מהירות ה-`Selective search` הופך לצואר בקבוק.



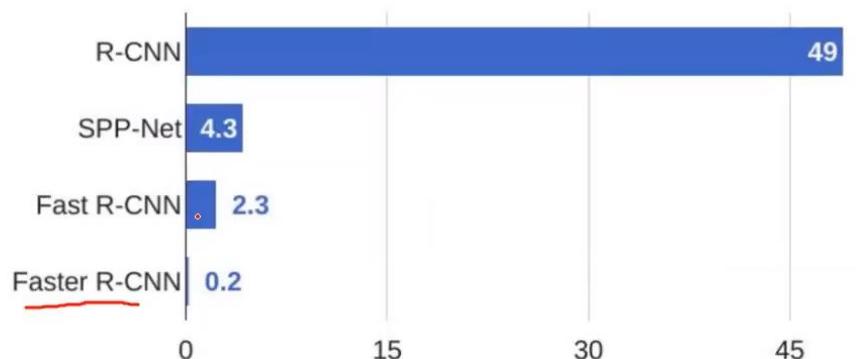
Faster R-CNN

از אחרי Fast R-CNN החוקרם הבינו שהצואר בקבוק החדש שלו הוא בעצם הצעה של אזור העניין, ומה יותר טוב מלהפוך גם את חלקו בתוך הרשות שלנו או אם נרצה לרשף בתוך רשות, ועכשו יש לנו רשות שמייצרת לנו את אזורי העניין (Region Proposal) (Region Proposal)



از בעצם אנחנו רואים שיש לנו רשות אחת שמייצרת הצעות לאזורי עניין, הצעות אלה עברת RoI pooling, ולאחר מכן אנחנו בודקים loss classification loss ו- Bounding-box regression loss ותכלנו את כל התהילה לתוך רשות אחת ומכובן שנכפה שהטהילה יהיה הרבה, ובמהירות הכוונה להרבה יותר מהיר.

R-CNN Test-Time Speed



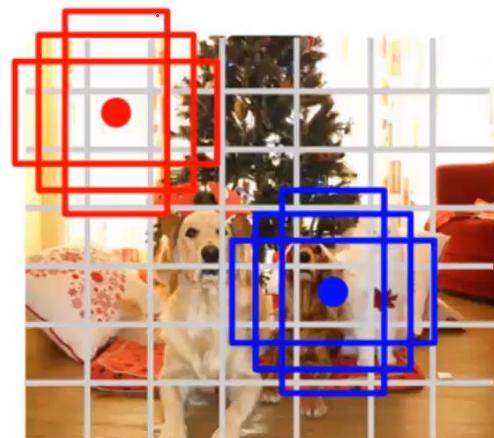
מה שנחננו רואים פה זה באמת נחדר, אבל זה עדין לא נותן לנו פיתרון של real-time כי אם בצפיה של וידאו אנחנו מגיעים ל-24 פרייםים בשניה, פה אנחנו עדין רחוקים מזהה. אז איך כן ניתן לשפר את התהיליך זהה?

התשובה נתנה באמצעות ארכיטקטורה שנקראת **YOLO**

YOLO – (you only live once)

הרעיון שעומד מאחורי YOLO במקומ לחיזות הצעות אזוריים שבמוסיפים הרצה של RoI pooling מלען רשות שכבר יצרנו או השתמשנו בה בשביל קבלת פיצ'ר מאפ, אנחנו מגדירים מראש עוגנים (anchors), הרעיון שאנו מיצרים מראש את כל אזורי העניין והדבר שיעזר לנו לצמצם את כל התוצאות המיותרים יהיה Non-Maximum suppression בו מילא אנחנו משתמשים בו בפרדיקציה אז הפעם נשתמש בו כבר בפעם הראשונה בשלב האימון.

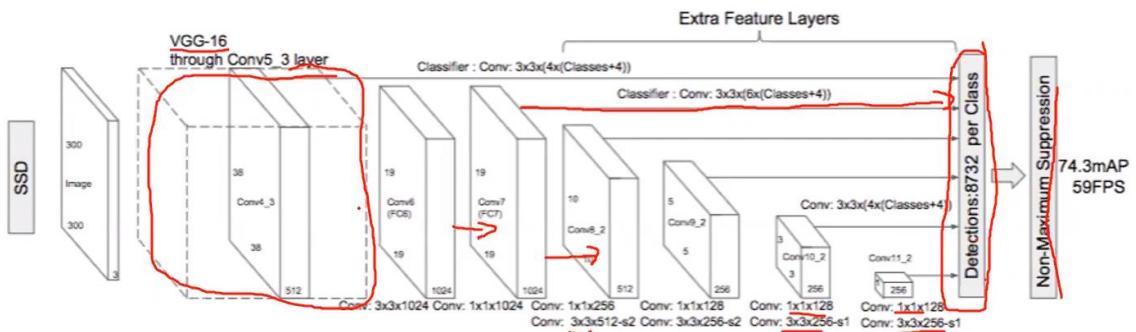
הרעיון הגדול שברגע שאנו מקבעים את אותו עוגן אז אנחנו יכולים לחשב גם את אותן פרדיקציות כחלק מאותה רשות.



SSD – Single Shot MultiBox Detector

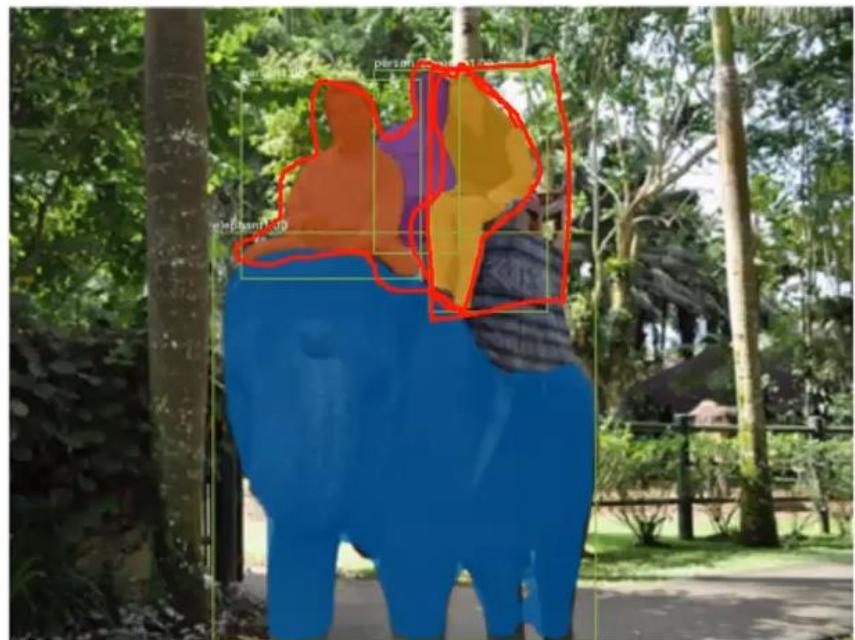
אנחנו מתחילה בארQUITטורה של VGG16 ותוצאה, את הפיצ'ר מאפ אנחנו מעבירים בשכבות קונבלוציה נוספת, שבסיום נסתכל על פרדיקציות של כל אותן שכבות נוספות (YOLO).

את כל אותן פרדיקציות נעביר דרך NMS.

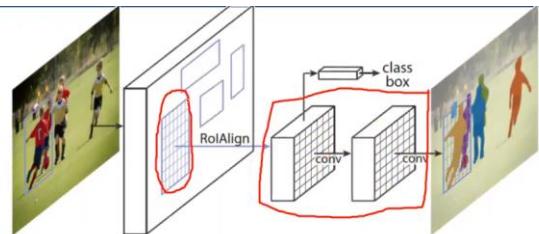


Putting it all together – Mask R-CNN

משלב את העולם של הקלסיפיקציה ברמת הפיישל יחד עם הרעיון של אובייקט דיטקיין, הרעיון בעצם אומר שקדם כל מיצר דיטקיין (באונדיניג בוקס) ולאחר מכן איזה חלק מתוך האובייקט שנמצא באותה קופסה איזה תחום שייך לאובייקט העניין שלו.



- Add a branch for predicting the class for each pixel in relevant regions

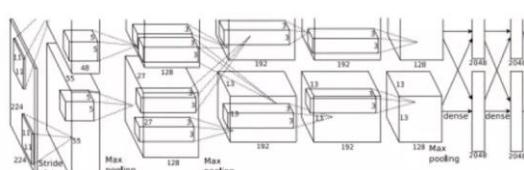
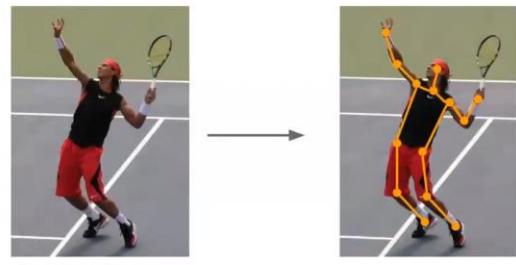


מציאת חלקים שונים באובייקטים שונים

Pose Estimation

- Represent pose as a set of 14 joint positions:

Left / right foot
 Left / right knee
 Left / right hip
 Left / right shoulder
 Left / right elbow
 Left / right hand
 Neck
 Head top
 ...



→ **Left foot:** (x, y)
 → **Right foot:** (x, y)
Vector:
 4096
 → ...
 → **Head top:** (x, y)

הרצאה על auto encoders

אז מה זה בעצם auto encoder?

אם עד עכשוו דיברנו על רשותות שהמטרה שלhn היא ללמידה סוג של מיפוי מרחב הקלט לאיזשהו פלט רצוי, בזאת auto encoder זה הפעם הראשונה שנטען בוג'ינג supervised learning, כלומר למידה שהיא לא למול איזשהו target רצוי, וזה בשונה ממה שהזכרנו בשיעור שעבר שבעם דיברנו על self-supervised learning.

מה ההבדל בניהם בעצם?

ב- self-supervised אנחנו משתמשים בתנאי הקלט עצמו כדי ליצור איזשהו תיוג רצוי.

ב- unsupervised אנחנו לא מנסים למדוד איזשהו תיוג רצוי, אנחנו רוצים למדוד דפוסים בתוך המידע שלנו.

בגדי אפשר להסתכל על self-supervised כמקרה פרטי של unsupervised learning שבו אנחנו משתמשים במקרה מסוים בתנאי הקלט שלנו בתיאור תיוג, או מבצעים איזשהי מניפולציה על נתוני הקלט שלנו בצורה שمدמה מצב של supervised learning.

ב-self encoders אנחנו בעצם מדברים על supervised learning, אנחנו לא מנסים למדוד משאו קבוע מראש (תיוג קבוע מראש), אנחנו בעצם מנסים למדוד מיפוי מהקלט אל הפלט עצמו.

השאלה שנסאלת היא בעצם למה בכלל שנרצה למדוד צזה מיפוי? למה לי למפות את הקלט לעצמו? מה אני יכול להרוויח?

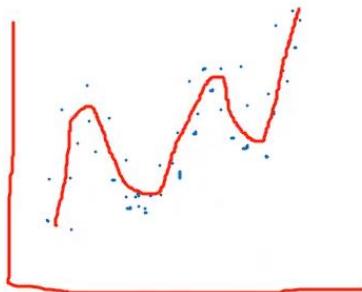
- ניתן לעשות קומפרסיה למידע מסוים (דחיסה)
- יציג חדש של המידע עם איזה שילוב של הורדת מדדיות ובשאיפה שהמידע יהיה גם אינפורטטיבי יותר.(representation learning).

נשים לב שההנחה הבסיסית ב-self encoders שיש דפוסים בתחום הנתונים, וכך שהדפוסים האלה יופיעו את הנתונים שלנו בצורה משמעותית יותר (הכוונה לכך לכמות הרעש בתחום), היכולה שלנו לייצר auto encoders שיופיעו ממשמעותיים ו שימושיים تعالה, וכך שהנתונים יהיו בלתי תלויים גם בין דוגמאות וגם בתחום פיצרים לא נוכל למדוד יציג חדש טוב, מכיוון שהנתונים לא תלויים אחד בשני ואז למידת תבניות לא תעוזר לנו.

Common use cases for Auto Encoders

- דחיסת המידע – Compression

- Anomaly detection – אם אני יכול ליזג או ללמד יציג מהנתונים שהם שמתנהגים באופן רגיל, כשאני לא אצליח לשחרר דוגמה מסוימת מtower הנתונים יכול להיות מאוד שזה יאמר על אותה דוגמה שהיא אונומאלית, ביחס לשאר הנתונים או ביחס למרבית הנתונים.

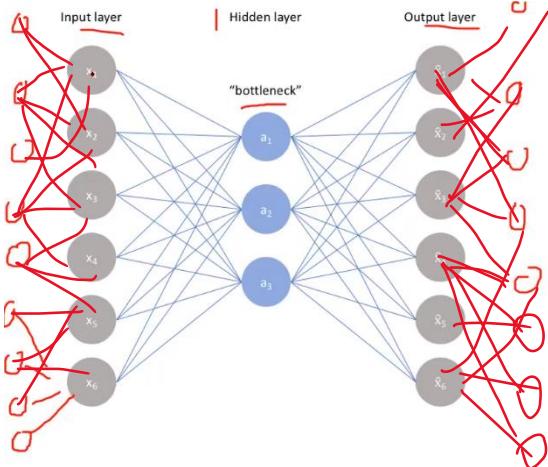


- Denoising – ניקוי רעש או הפחחת רעש מתוך נתונים. מטרה נוספת יכולה להיות הבנה בעזרת רעש כיצד סיגナル יכול להיראות ובעזרת זה ללמד משהו על דפוס הנתונים.

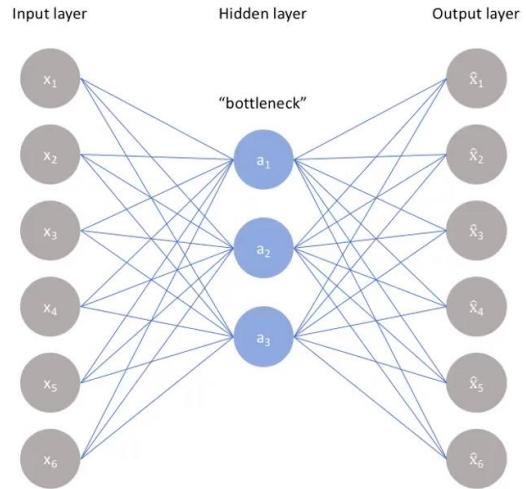
- Representation learning – יכולים ללמד יציג לשם למידת היציג, כשהשימוש ביציג זה יכולם להיות מגוונים, יכול להיות לטובת הוצאה פיצרים לתהילך אחר, יכול להיות לאפיון מקדים לפני סגמנטציה, יכול להיות שימוש לכל אלגוריתם אחר יוכל ממש לקבל יציג יותר דחוס או מפורט או מתאים לשימוש מסוים מתוך שימוש בauto encoder.

אין נראה :auto encoder

Deep auto encoder (more hidden layers)



Classic auto encoder



Bottle-neck – השכבה הצרה ביבוע והמרכזית, בדרך כלל נשתמש בה ל טובת הייצוג החדש שאנו לומדים, לאחר מכן אנו לומדים משקלות חדשות שלמדו אותנו לחזור מאותו bottle-neck representation לאוטו output שהוא latent representation שלנו.

כל שגדיל את הממד של ה-bottle-neck הבנייה מחדש בזורטא reconstruction error שהול וקטן, זה יכול להיות מוסבר בצורה פשוטה, שב עצמי כמות קטנה של פיצרים מסה להסביר איך הוא קלט מאד מורכב. (אם יש פה ??trade off)

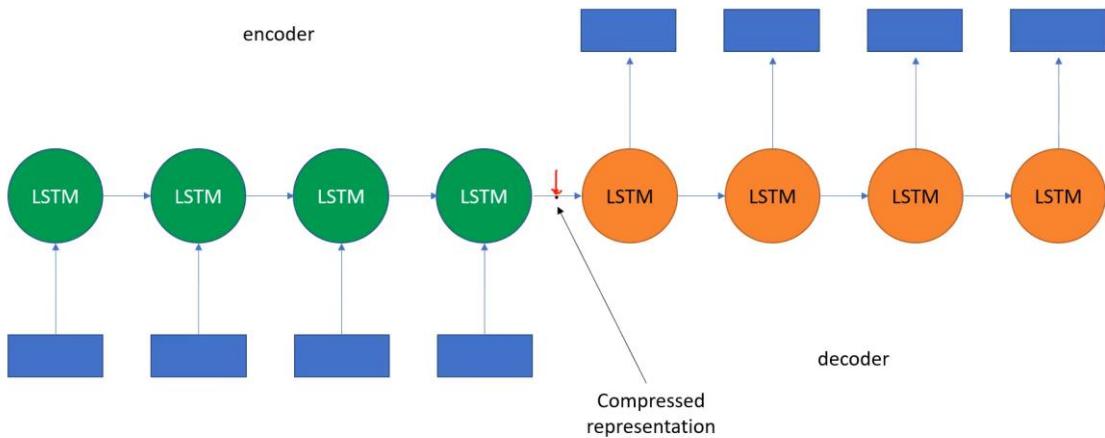
LSTM Auto Encoder

שימוש של LSTM בתורת auto encoder

מה זה אומר?

בצם אנחנו מנסים למפות סט נתונים לאותו סט נתונים עצמו. בשלב הראשון בעצם מעבירים את state בין ה-LSTM השונים ומהנקודה של Compressed representation אנחנו מנסים לשחזר את הסיגナル המקורי בשלב אחרי שלב.

החלק הראשון הוא encoder והחלק השני הוא בעצם decoder



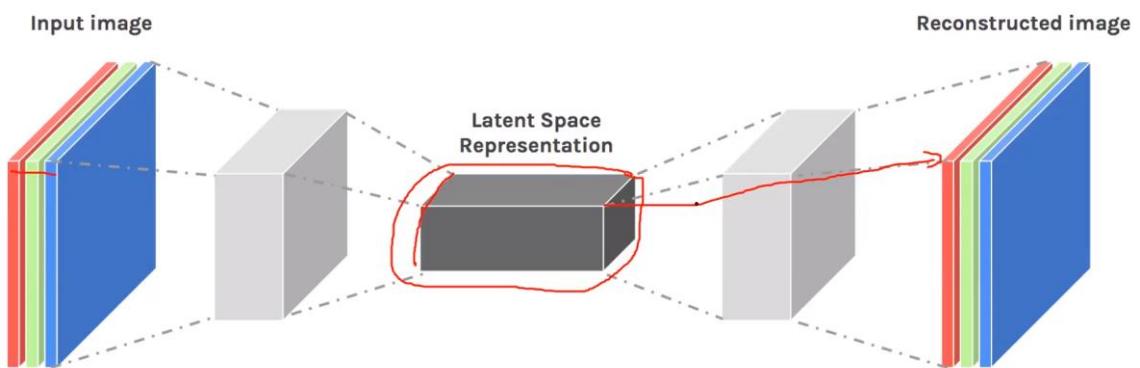
מתאים לו:

- Denoising
- זיהוי אונומליות – אם אנחנו מצלחים להגיע לאיזשהו state שמייצג את כל מה שראינו עד עכשיו ובהנחה שה loss reconstruction הוא נמוך עבור נקודות שמתנהגות בaczera קבועה(באייזשהו דפו שוחרר על עצמו), כשאנו מקבל דוגמה אונומלית אז הרatio loss reconstruction שלנו יהיה גבוה, כלומר ההבדל בין הoutput לoutput יהיה גבוה.

CNN Auto Encoder

יש לנו אפשרות להשתמש ב-CNN auto encoder, שמה שאנו לומדים כאן זה איזשהו Latent Space שהוא תלת ממדי שבעצמו מכיל את כל מפות הפיצ'רים של ממדנו ומאותן מפות פיצ'רים אנחנו יכולים לעשות reconstruction לפلت המקורי.

מציר סיטואציה של Super-Resolution שבה בשלב decoding אנחנו לומדים את המעבר ממדים מצומצמים/דחוסים לאיזשהו ממד יותר מפורט, ההבדל שכן נוכל לקבל יותר מפות פיצ'רים מאשר שלושת העורצים שקיבלו בזקוקו, לעומת זאת אנחנו יכולים להחליט אנחנו מגבלים את עצמו לאיזה input (ב- latent Space) מצומצם יותר.



GAN – Generative Adversarial Networks

הרעיון של GAN – אם עד עכšíו היה לנו איזשהו מודל Shmkeyl training data ובהתאם לכך יכלנו לייצר מודל קלסיפיקציה שאומר אם נתונים שייכים לקטgorיה אחת או לקטgorיה אחרת.

אנחנו גם יודעים לייצר מודל שמייצר נתונים בהינתן קלט מסוים, הקלט יכול להיות low resolution image והוא יכול להיות גם סוג של רעש (בහינתן רעש מסוים, לייצר משהו שmbased על הרעש, כמו בדוג' עם פונקציית סינוס שראינו בקורס בכיתה).

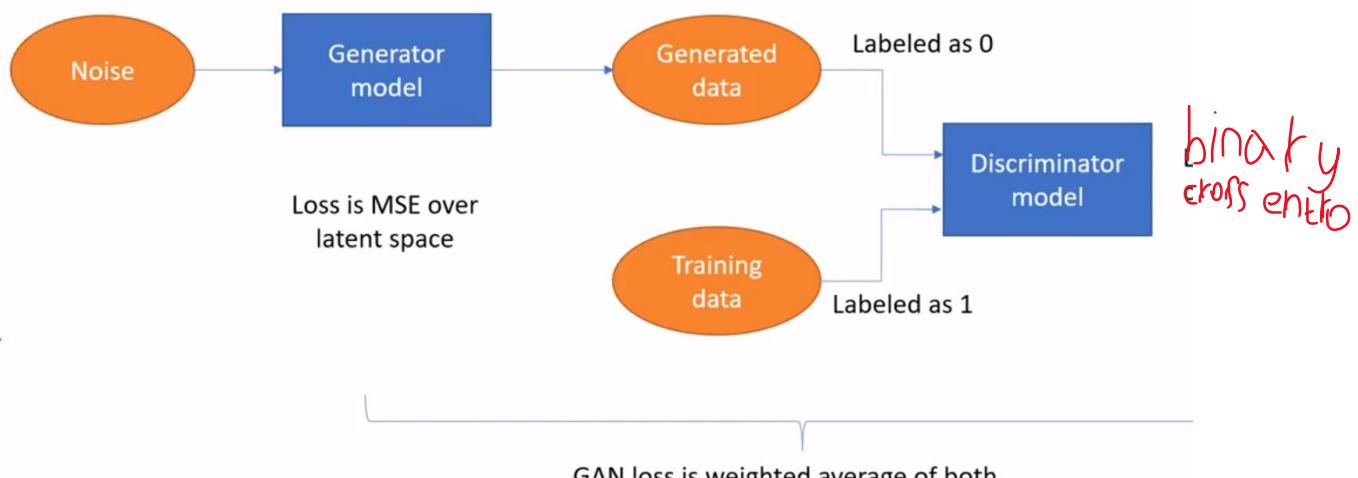
ואז הגיע המחשבה למה לא לשלב את שני סוגי המודלים האלה?

כלומר, מודל קלסיפיקציה שאומר אם דוגמה מסוימת שייכת אליה אם היא אובייקט אמיתי, או אינה שייכת מגנרטר (Generator) שלא הגיע ממהולם האמיתי.

אם אני יודע לעשות הבחנה כזו באמצעות קלסיפיר מבוסס רשת ניורונים, ואני יודע לייצר מודל שмагנרט נתונים בהתאם איזשהו רעש או קלט מסוים, אני יכול לאפטם (אופטימיזציה) את שניהם יחד בצורה כזו שה-loss binary cross entropy loss,binary cross entropy loss במודל הגנרטור שהוא משמש שמתאים לבעה (נגיד עברו בעית חולוציה זה יהיה MSE או PNSR) ויחד תוכל להרכיב loss שמתכלי את שניהם.

מצד אחד נרצה שה-loss Discriminator model (המודל קלסיפיקציה) יבדיל בין דוגמאות אם הן הגיעו מהנתונים שאחנו מתאמנים עליהם או מנתונים מגנרטים, ומצד שני נרצה שהגנרטור יצליח לשוטות באוטו מודל Discriminator כמה שאני יכול כשאני מאפטם את המשקولات של הגנרטור ואני רוצה להצליח להבדיל האם נתונים הגיעו מ-data מקורי כשהאני מאפטם את המשקولات של ה-

.Discriminator



ה-loss של כל GAN מורכב מהloss Discriminator loss ומה-loss generator, והחישוב שלו הוא בעצם משקל ממוצע של שניהם.

שימושים של Deep Learning בתחום השוניים

:Visual domain

- Classification – קלסיפיקציה מחלות, חיפוש מבוסס תמונה, קלסיפיקציה פעולות (לפי תמונה או סרטון), מציאת דמיון בין מוצרים.
- Segmentation – זיהוי לאובייקטים מסווגים בתוך תמונה, ספירת אנשים בקהל וכו'.
- Detection – חיפוש אלמנטים בתוך תמונה, ספירת מוסדות לענות על שאלת, לתאר במשפט מה קורה בתמונה, תרגום טקסט מתוך תמונה או הוצאה של טקסט מתוך תמונה.
- Image-text cross learning – בהינתן תמונה מסוימת לענות על שאלת, לתאר במשפט מה צביעת תמונה, דיפ פיס, זיהוי עומק בתמונה, סופר רחולציה.
- Generation (GANs) – שערוך מוצר/נדל", זיהוי חומרה של מחלת לפי תמונה.
- Regression

:Language domain

- האם משפט מסוים בוטה מדי, האם יש חיוביות או שליליות במשפט. Classification – part of speech tagging, Semantic role labelling.
- Machine Translation – תרגום שפות
- Summarization – סיכום מאמרם מדיעים (היכולת לתרמצת חמורים), הרבה פעילות מחקרית סביב עולם המשפט (כמו מציאת אוטומטית של אלמנטים מפלילים).
- Sentiment analysis – זיהוי במרקם שירות, האם מצב רוח הלגוך טוב או לא Question Answering – בוטים שעונים על שאלות (ChatGPT).

:Other domains

- Sound Separation – היכולת לתרמצת שיחה למשימות או מי אמר מה וכו'
- Time series – זיהוי אונומליות, signal disaggregation (מידע שמתככל הרבה סיגנלים ונרצה לפצל אותו למקורות השונים שלו, למשל בהינתן צリכה כוללת של חשמל איזה מכשירים הביאו אותה צリכה כוללת של חשמל), קלסיפיקציה (זיהוי פעולות שונות לפי TS), סגמנטציה וחיזוי.
- Tabular Data – נושא שהוא פחות מדובר בעולם של הלמידה عمוקה אבל גם שם יש פיתוחים כמו המלצות תוכן ולפעמים גם בעולמות אחרים.
- Graph based learning – היכולת לתיכל נתונים מתוך גרף ידוע, למשל גרפ כל הקשרים בין ישות בוויקיפדיה, יכול לומר על ישותות חדשם.