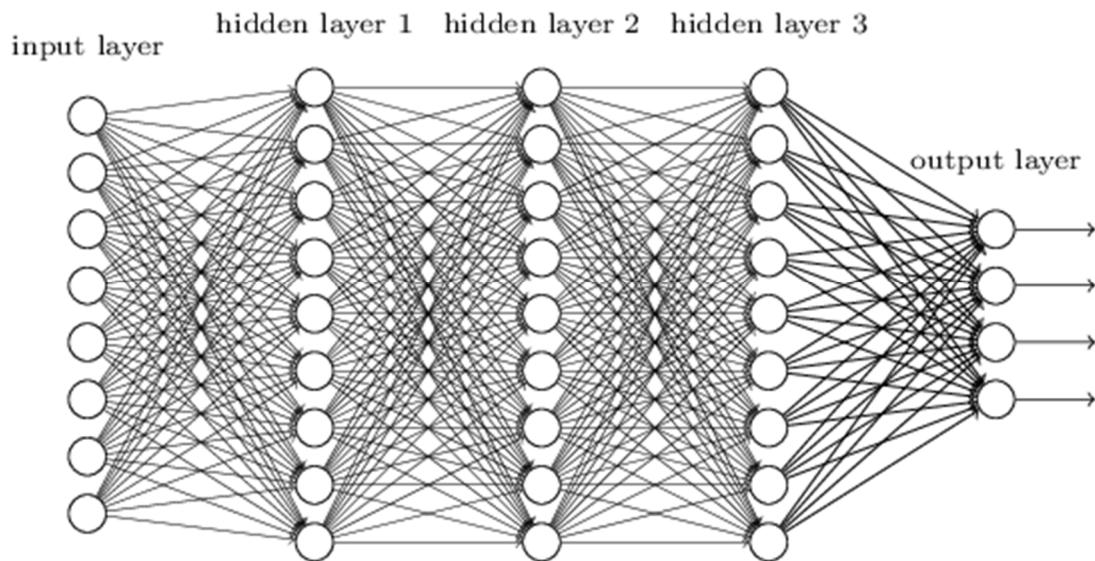




ניסוי מעבדות 3-2

מבוא ללמידה عمוקה

Introduction to Deep Learning



כתב על ידי : נדב בהונקר ושותנית חביב

עודכן בתאריך : ספטמבר 2018
יולי 2019 (אורן בריט)

תוכן עניינים

פרק למעבדה – מפגש ראשון.....	4
מטרת הניסוי.....	4
פרק על מערכות לומדות.....	4
בעיית הסיווג.....	4
1. דוגמה פשוטה.....	4
2. הגדירות.....	5
3. מודד ביצועים.....	9
4. תהליך התכנן של המסוווג.....	9
Gradient Descent.....	10
רשתות נוירונים.....	11
אלגוריתם האימון של רשת נוירונים.....	13
סיווג באמצעות שגיאה ריבועית ונויירון בודד - ADALINE.....	13
1. אימון באמצעות Gradient Descent.....	15
סימולציה - בעיית סיווג האירוסים.....	17
גרסיה לוגיסטיבית.....	17
חישוב הגרדיינט.....	19
אלגוריתם אימון וגרסיה לוגיסטיבית.....	20
גרסיה לוגיסטיבית כמודל שכבות.....	20
בעיית סיווג מתוך מחלקות רבות.....	22
לקראיה נוספת.....	23
מפגש ראשון.....	24
שאלות הינה.....	24
מפגש ראשון - מהלך הניסוי.....	26
הנחיות.....	26
חלק א' – גרסיה לוגיסטיבית.....	27
חלק ב' – זיהוי ספרות בתמונה (סיווג בעזרת רשת נוירונים).....	28
חלק ג' – סיווג ספרות בעזרת Matlab Neural Network Toolbox.....	30
פרק למעבדה – מפגש שני.....	33
הקדמה.....	33

33.....	מבוא
34.....	רשתות קונבולוציה
37.....	שכבות הורדת מים
37.....	סוגי אי-lienאריות
39.....	שכבות שארית (Residual)
40.....	רשתות CNN סטנדרטיות
41.....	שיטות אופטימיזציה מתקדמות
41.....	1. מומנטום
42.....	2. Adaptive Gradient (AdaGrad)
42.....	3. RMSprop
43.....	4. Adaptive Moment Estimation (Adam)
43.....	רגולריזציה
43.....	.1 Early Stopping
44.....	.2 פחות פרטיטרים
44.....	.3 Weight Decay
44.....	.4 Dropout
45.....	.5 Data Augmentation
45.....	מה נלמד בשכבות הפנימיות בראש
47.....	Transfer learning
48.....	מפגש שני
48.....	שאלות הינה
49.....	מפגש שני – מהלך הניסוי
49.....	חלק א' – סיווג תמונות CIFAR10
51.....	חלק ב' – Transfer Learning –
53.....	נספח א' – Matlab Layers חלק א'
54.....	נספח ב' – Matlab Layers חלק ב'

רקע למעבדה – מפגש ראשון

מטרת הניסוי

היכרות עם אלגוריתמי למידה חישובית משפחחת רשותת הנוירונים. אלגוריתמים אלה הוכיחו את עצם בשנים האחרונות כבעלי ביצועים גבוהים וכעת הם משמשים ככלי יישומי בהרבה תחומיים ותעשייה. בניסוי נבנה את היסודות כדי להבין את כיצד האלגוריתם פועל ונמשך רשת נוירונים. מכיוון שרשותת נוירונים הינה אלגוריתמים בתוך התחום הנקרא מערכות לומדות, יש צורך בידע של מושגים בסיסיים בתחום.

רקע על מערכות לומדות¹

מערכות לומדות (Machine Learning) הוא תחום העוסק בפיתוח ותכנון אלגוריתמים המאפשרים מיצוי אוטומטי של מידע מתוך נתונים אמפיריים. לפי אחת ההגדרות, מערכת לומדת היא מערכת אשר משפרת את ביצועה בбиוץ מושימה נתונה ככל שהיא מבצעת מושימה זו. בתחום יישומים רבים ומגוונים: זיהוי כתוב יד ודיבור, סיוג מסמכים, לימוד במשחקים, תרגום, רובוטיקה, זיהוי פנים ועצמים בתמונה, חיזוי פיננסי ועוד. בנגד לפתרון אלגוריתמי "מסורתי", בו האלגוריתם מפורש וקבוע וכל פרטיו הפתרון ידועים למתכנן, אלגוריתם לומד מכתב עד כדי מאפיינים תלוי מידע, המכוננים במהלך הלימוד, לגישה לומדת יתרונות רבים, ביניהם הקניית יכולות שחן מעבר ליכולת הניתוח של מפתח המערכת, והסתגלות לשביבה משתנה. כמוון שיחד עם היתרונות הרבים קיימים חסרון, ביןיהם הצורך בכמות גדולה של מידע מותוו באופן ספציפי ואיכותי וכוח החישוב היקר הנדרש.

בעיית הסיוג

נציג כעת את בעיית הסיוג תוך התיאchorות לדוגמה:

1. דוגמה פשוטה

במפעלי אריזות דגים מעוניינים להפריד באופן אוטומטי בין הדגה היומית. בפרט, יש להפריד בין דגי הסלמון (salmon) לדגי הלבrex (sea bass) על סמך תמונה של הדג, דהיינו למצוא מסווג שמוסיפה עבור כל תמונה פלט המציין את סוג הדג. למשימה מסווג זה קודם של מיצוי מאפיינים "מעוניינים" מתוך התמונה. מאפיינים אלו יסייעו לנו בסיווג הדגים. הוחלט כי ימוצאו מתוך התמונה שני המאפיינים הבאים: אורך הדג ובהירות הדג (בhinintן שהתמונה נתונה ברמות אפור). נציגו שמיוצוי המאפיינים מתוך תמונה דורש הפעלת עיבוד תמונה על-מנת לנוקות את התמונה מרעש ולבצע סגמנטציה של הדג מתוך הרקע. במעבדה זו לא עוסוק בשלב זה ונניח כי בידינו שני המאפיינים הרצויים עבור כל תמונה.

¹ החומר מתבסס על הרצאה לניסוי "מבוא למערכות לומדות" של המעבדה לבקרה, רובוטיקה ולמידה חישובית.

2. הגדירות

בבואה הסיווג אנו נדרשים לתכנן מסובג באמצעות סט דוגמאות מתוויות כך שיסובג בצורה הטובה ביותר קלט חדש.

נשתמש בהגדירות הבאות לתיאור בעית הלמידה :

- מרחב הקלט : $\mathbb{R}^D \in X$, כך שכל דוגמה $X \in \mathcal{X}$ מוגדרת כ- $x = (x^1, x^2, \dots, x^D)$. כאשר $D > 1$, נקרא ל- x "וקטור המאפיינים".
- בדוגמה : $\mathbb{R}^2 \in X$, כך שבעור כל דוגמה $X \in \mathcal{X}$ יש שני מאפיינים : אורך ובהירות.
- מרחב הפלט : $\Omega = \{1, 2, \dots, C\}$ מכיל את אוסף המחלקות האפשריות.
- בדוגמה : $\Omega = \{1, 2, \dots, C\}$ מכיל במקרה זה שתי מחלקות לסיווג: סלמוני ולברך. בעית סיווג מסווג זה, עם שתי מחלקות בלבד, נראית בעית סיווג ביןארית.
- מושא : העתקה $\Omega \rightarrow \mathcal{X}$ אשר נותנת לכל קלט במרחב הקלט תיוג.
- הפונקציה אותה נרצה ללמד היא זו המתוימת כל תמונה אם מדובר בזג סלמוני או זג הלברך.
- סדרת הלימוד (training set) : סט של n דוגמאות מתוויות (labeled) $\{(x_i, y_i)\}_{i=1}^n$, אשר $y_i \in \Omega$ הוא הסיווג הנוכחי של תבנית הקלט.
- בדוגמה : תמונות דגים אשר מחלקו תיוגה באופן ידני.
- סדרת הבוחן (test set) : סט של m דוגמאות (שאינו שיך לשדרת הלימוד) $\{(x_i, y_i)\}_{i=n+1}^{n+m}$. סט זה משמש להערכת הביצועים של המסובג הנלמד. בשלב הערכת הביצועים נפעיל את המסובג על וקטורי המאפיינים $\{x_i\}_{i=n+1}^{n+m}$ ונשווה את התוצאות של המסובג לתוצאות הנוכחיים $\{y_i\}_{i=n+1}^{n+m}$.
- בדוגמה : תמונות נוספות של דגים שתויגו ידנית בהם לא השתמשנו באימון המסובג.

באופן כללי, נהוג לחלק את בעית הלמידה לשניים : בעיות סיווג ובעיות רגרסיה. ההבדל היחידי בין השניים הוא מרחב הפלט. בעיות סיווג הוא בדיד, ובבעיות רגרסיה הוא רציף. לדוגמה, בעית סיווג הדגים המחלקות הן הדגים: סלמוני או לברך. לו היינו מנסים לקבוע את משקל הזג מתוך תמונה, הרי זו הייתה בעית רגרסיה, שכן הפלט הוא מספר רציף.

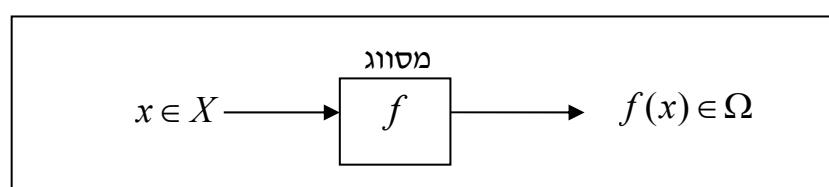
אלגוריתמים רבים ללמידה בעיות סיווג ניתנים להמיר לצורה פשוטה בעיות רגרסיה, וכן כל העקרונות שילמדו במעבדה זו לגבי רשותות נוירונים תקפים לגבי בעיות רגרסיה.

לרוב נניח כי סדרת הלימוד וסדרת הבוחן שלנו הן בלוטות אינטגרליות ובלתי תלויות (Independent and identically distributed random variables) *i.i.d.* כלומר :

$$p(x_i) = p(x_j), \quad \forall x_i, x_j \in \{\text{training} \setminus \text{test set}\}$$

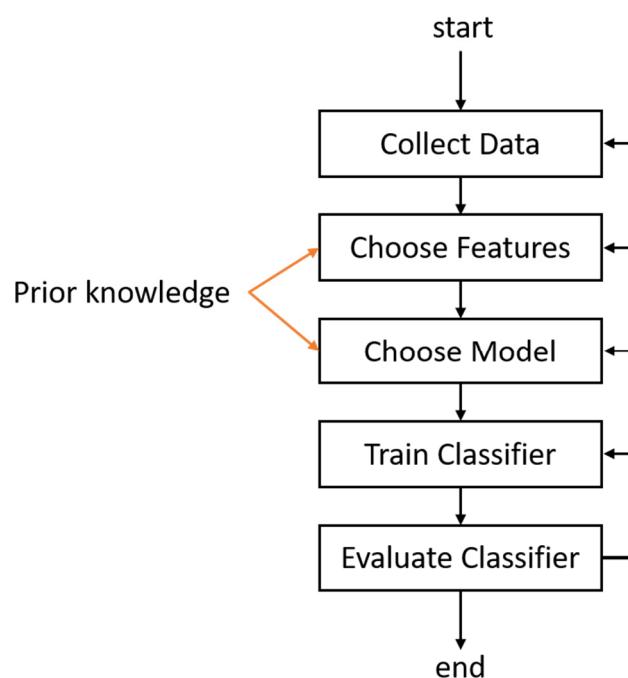
$$p(x_i, x_j) = p(x_i)p(x_j), \quad \forall x_i, x_j \in \{\text{training} \setminus \text{test set}\}$$

נגיד רשות את בעיית הסיווג תוך שימוש במושגים הנ"ל:
 בהינתן סדרת לימוד $\{x_i, y_i\}_{i=1}^n$, נרצה למצוא מסובג $\Omega \rightarrow f : X$ כך שיסווג את סדרת הבוחן $\{x_k\}_{k=n+1}^{n+m}$ למחלקה המתאימה עם שגיאה קטנה ככל הניתן.
 באופן סכמטי מסובג מוגדר בצורה הבא:



הלמידה המתוירת הינה למידה אינדוקטיבית: הכללה מהפרט - סדרת הלימוד, אל הכלל – קלט חדש. בפרט נשים לב כי נדרש לתכנן מסובג בעל שגיאה קטנה על דוגמאות חדשות שלא שייכות לסת הדוגמאות בו נעזרנו לתכנון.

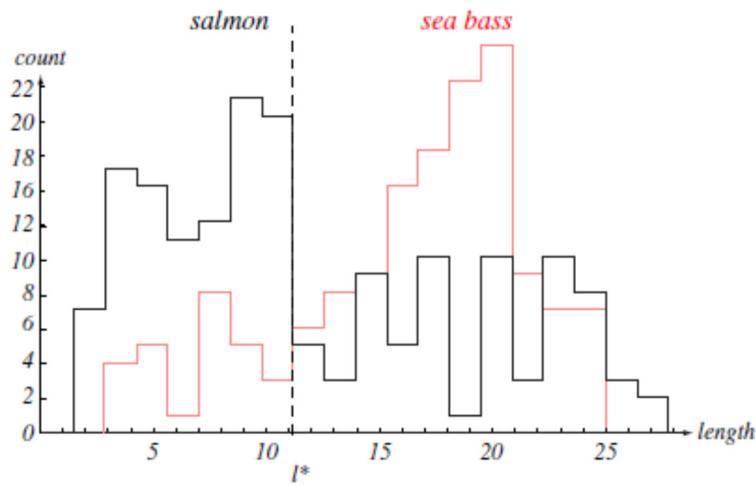
התרשימים הבא מתאר בצורה סכמטית את תהליך הלימוד בבעיית הסיווג.



איור 1 – תיאור סכמטי של תהליך הלימוד

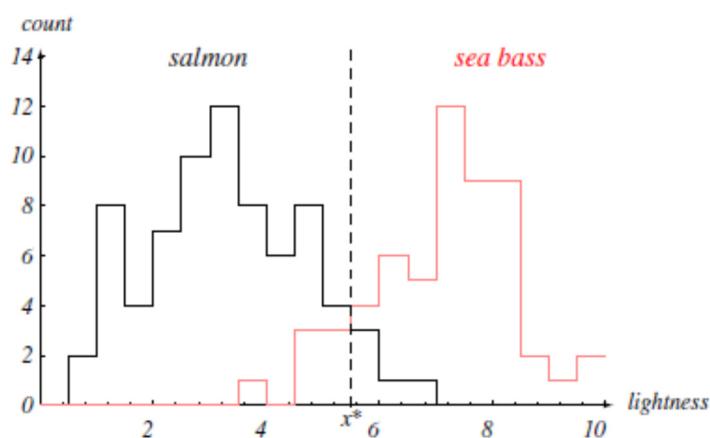
לאחר שלב איסוף המידע (תמונהות של דגמים עם תיוג), יש לבחור את המאפיינים הרלוונטיים לтиיארו ואת המודל המתאים למערכת (אורך ובהירות); בשלב זה ניתן לשלב ידע מקדים אודות המערכת. לאחר מכן מגיעו שלב אימון המסובג ובחינת ביצועו. הידע המקדים אותו אנחנו מכנים למערכת בדוגמה הדגים הינה שניתן להסתפק בהירות ואורך הדג על מנת לקבוע את סוגו.

בשלב ראשון נtabונן בדוגמאות שלנו, ובהתפלגות המאפיינים השונים. באירור 2 מוצגת ההיסטוגרמה של הדוגמאות ע"פ המאפיין של אורך הדג². ניתן לראות כי אין הפרדה טובה בין המחלקות מכיוון שהחפיפה גדלה מידי.



אייר 2 – ההיסטוגרמה ע"פ אורך הדג

באיור 3 ניתן לראות את ההיסטוגרמה של הדוגמאות ע"פ המאפיין של בהירות הדג. כאן תחום החפיפה קטן יותר שכן ברוב המקורים הלבך בהיר יותר.

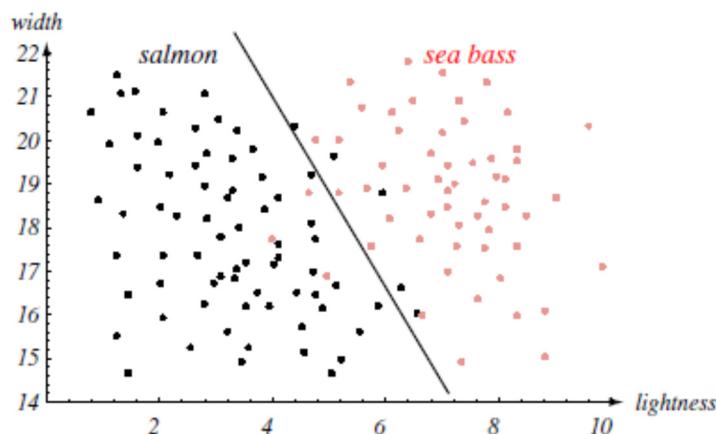


אייר 3 – ההיסטוגרמה ע"פ בהירות הדג

ככל האירורים מלקחו מຕוך :

Pattern Classification (2nd ed.), Richard O. Duda, Peter E. Hart and David G. Stork (John Wiley and Sons, 2001)

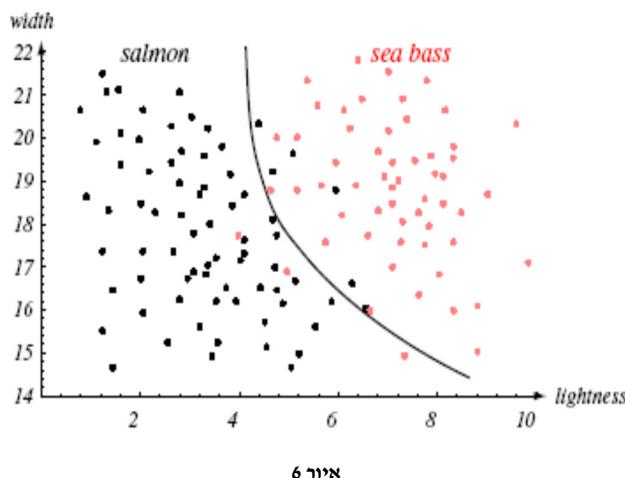
נשאלת השאלה האם שימוש בשני המאפיינים יחד יכול להביא לייצוג טוב יותר של הבעיה. על מנת לענות על שאלת זו, ניתן לציר את ערך המאפיינים בתרשים דו-ממדי, כפי שモצג באירוע 4. כפי שנייתנו לראות, ניתן להבהיר קו ישר בין שתי המחלקות המסוג באופן נכון את רוב דוגמאות סדרת הלימוד. ניתן להבחן שבאזור היצוג הדו-ממדי שתי המחלקות ניתנות להפרדה טובה יותר מאשר בשימוש באחד מהמאפיינים.



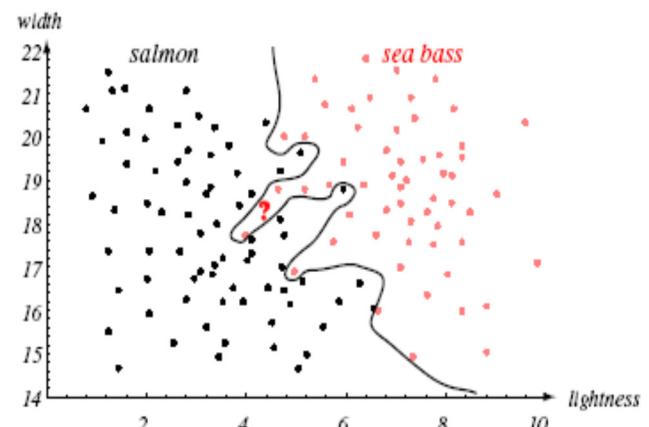
אירוע 4 – הצגה דו-ממדית של המאפיינים

כמובן שקיימות אפשרויות נוספות להעברת הקו לשילוב המחלקות. שתי דוגמאות מוצגות בהמשך. באירוע 5 ניתן להבחין כי מושג סיווג מושלם על סדרת האימון אך אזור ההחלטה מאד מסובכים. אזור ההחלטה כאלו יכולים להשיג אפס שגיאה על סדרת האימון, אך עשויים להוביל לשגיאה גדולה יותר על דוגמאות חדשות. באירוע 6 מוצג מסוג בעל איזור החלטה עם פחות שגיאות על סדרת האימון מזו שבאירוע 4, אך מעט יותר מסובך.

בתכנון מסווג יש להתייחס ל tradeoff בין סיבוכיות המודל והביצועים על סדרת האימון.



אירוע 6



אירוע 5

התופעה המתוארת באירוע 5 היא תופעה נפוצה ומורכמת כאשר משתמשים במסוגים מורכבים בעלי פרמטרים רבים. תופעה זו נקראת overfitting, ויישן מספר אינדיקציות לקיומה בהינתן מסווג מסוימים.

3. מדריך ביצועים

נשתמש במדד ביצועים כדי לבדוק את מידת ההצלחה של המסובג שלנו f . נערך את שגיאת

המסובג באמצעות סדרת הבוחן (test set) $\{x_i, y_i\}_{i=n+1}^{n+m}$:

$$Err(f) = \frac{1}{n} \sum_{i=n+1}^m l(f(x_i), y_i)$$

כאשר ℓ הינה פונקציית השגיאה (נקראת zero-one loss):

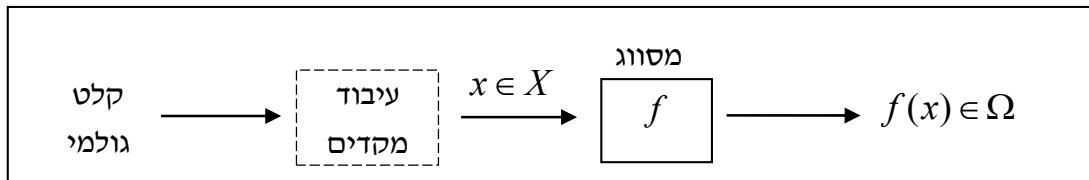
$$\ell(f(x_i), y_i) = \begin{cases} 0 & f(x_i) = y_i \\ 1 & f(x_i) \neq y_i \end{cases}$$

שגיאת המסובג הינה השגיאה האמפירית על סדרת הבוחן. נרצה למדוד f כזה ש יהיה קטן ככל האפשר.

4. תהליכי התכנון של המסובג

אלגוריתם הסיווג אינו מופעל בדרך כלל על הקלט הגולמי. קלט זה עובר שלבים של עיבוד מקדים לפני למידת המסובג והפעלתו.

באופן סכמטי, נוסיף את השלב של העיבוד המקדים לתכנון המסובג:



בדוגמה הדגים לעיל, חלק מהעיבוד המקדים היה שלב מיצוי המאפיינים, אורך ובחרות הדג, מתוך התמונה. שלב העיבוד המקדים נעשה בהתאם לאופי הקלט הגולמי ולסוג אלגוריתם הלמידה (אופי המסובג).

עיבוד מקדים של קבוצת הלימוד חיוני לטובת:

1. התאמת הקלט למודל הלמידה. כלומר, ישנו סוגים מסווגים המתאימים לקלט מסוים.
2. האצת שלב הלמידה או האימון של המסובג.
3. שיפור רמת הביצועים של המסובג.

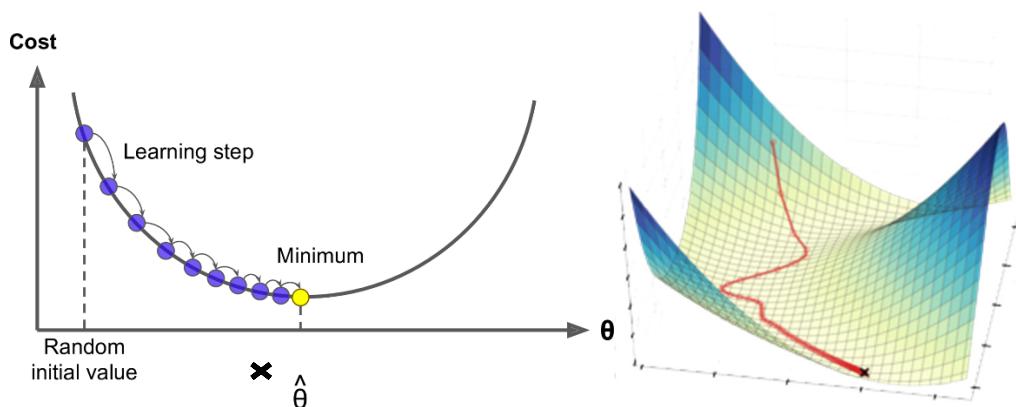
Gradient Descent

Gradient descent הינה שיטה כללית, איטרטיבית למציאת מינימום או מקסימום של פונקציה. הרעיון מאד פשוט: בהינתן פונקציה גזירה ($f(x)$, אותה נרצה למקסימולمزער, נעדכן את x לפי הנוסחה:

$$x_{k+1} = x_k \pm \eta_k g_k$$

$$g_k = f'(x_k)$$

כלומר, נזוז את הערך של x בכיוון בו הגרדיינט הוא מקסימלי (למינימיזציה- נגד כיוון הגרדיינט). כאשר η הינו מקדם הנקרא גודל הצעד או קצב הלמידה. גודל הצעד יכול להישאר קבוע או להשתנות בתהליך הלמידה. ויזואלית, התהליך עשוי להיראות כך:



איור 7 – המחשה ויזואלית של תהליכי אימון באמצעות אינטואיציה

ניתן להיעזרenganalogיה הבאה לקבלת אינטואיציה:
דמיינו את המצב ההיפותטי בו מטייל נמצא על הר בו שורר ערפל כבד כך שהוא לא מצליח לראות את תוואי השטח. האדם מנסה לדוד מההר (אל הנקודת הנמוכה). מכיוון שאין ראות, הדרכ ביהגיע למטרה היא ע"י בחינת הכיוון בו הירידה תלולה ביותר והליכה בכיוון זה מספר צעדים קבוע בקורס איטרטיבית.

enganalogיה:

- המיקום של המטייל מסומן ע"י x .
- תוואי השטח היא הפונקציה ($f(x)$) שורצים למזער.
- גודל הצעד של המטייל הוא גודל הצעד η .
- כיוון הירידה הוא גודל הגרדיינט.

הערות:

- כדי שנוכל להפעיל את האלגוריתם, כל שנדרש הוא שהפונקציה ($f(x)$) תהיה גזירה. אין מגבלה של מימד או צורה של פונקציה.

- האלגוריתם פשוט ביותר ביוטר למציאת מקסימום מינימום של פונקציה שומר על גודל צעד קבוע. ישן שיטות מתקדמות יותר : כאלו שמשנות את גודל הצעד בצורה אדפטיבית, כאלו שימושות בקירוב מסדר שני של הפונקציה (ההסיאן) ועוד.
- אין הבטחה שהאלגוריתם יכנס למקסימום מינימום הגלובלי ! יתרון מאוד שנכנס למינימום מקומי. הבטחת התכונות מתקיימת רק אם (x) f פונקציה קמורה.
- תחום המחבר של מציאת מקסימום מינימום של פונקציות נקרא אופטימיזציה וקיים מספר קורסים בטכניון בהם מלמדים את הנושא לעומק.

רשתות נוירוניים

כעת נציג את המודול הכללי של רשתות נוירוניים. המודול מתאר את (במיעט) כל רשתות הנוירוניים שקיימות. מהרשות הפחות ביוטר שנראה בהמשך, ועד הרשותות גדולות ומורכבות שימושות חברות מסחריות (זיהוי פראזופים, תרגום וכו').

למען ההקדמה, נסתכל על רשת נוירונים בעל קופסה שחורה. נרצה ש קופסה זו תהיה מסוגלת ללמידה מדוגמאות קיימות, ולאחר מכן להשתמש במידע שרכשה על מנת לבצע הסקה חדשה על דוגמה חדשה. מתאר את הכניסות והיציאות למערכת שלנו, ואת הפרמטרים הנלמדים :

כニיסות:

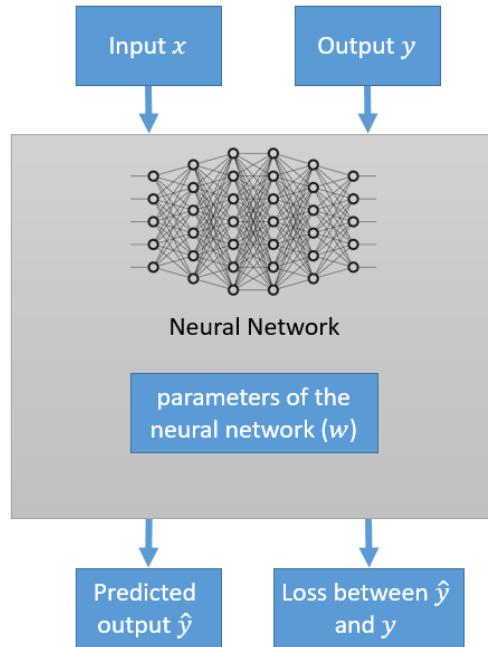
- x נקודות מידע כלשהי (תמונה, נתונים וכו')
- y תיוג מתאים לנקודה x

יציאות:

- \hat{y} חיזוי של תיוג מתאים לנקודה x
- l ה-Loss של החיזוי \hat{y} ביחס לתיוג האמתי y

פרמטרים נלמדים:

- פרמטרי רשת הנוירונים המשמשים לחיזוי התיוג. נסמנם ב- w , ונקרא w "ווקטור המשקלים".



איור 8 הממחשה ויזואלית של רשת נוירונים כקופסה שחורה. כניסה ויציאה

בשלב האימון של רשת הנוירונים, המערכת תקבל באופן סדרתי זוגות $\{y, x\}$, ותעדכן את הפרמטרים שלה w , על מנת להتاימים את המוצא החזוי \hat{y} להיות מתאים ככל הניתן לו. כלומר, נרצה לעדכן את פרמטרי המערכת במטרה **למזרע בכל הניתן את שגיאת החזוי**. לאחר שהמערכת אומנה, נרצה לבצע הסקה על דוגמה חדשה. נזין לכניסת המערכת את הדוגמה הרלוונטיות x , והמערכת תשתמש בפרמטרים אשר נלמדו בשלב הקודם על מנת לחזות \hat{y} מתאים. באופן זה ניתן לבצע חיזוי עבור דוגמאות שאנו לא יודעים את תיוגן, בעזרה למידה מדוגמאות מתוויות נתונות.

נוירוניים?

האלגוריתמים המכונים רשתות נוירונים במקור נוצרו בהשראת המוח. באיזה מובן? במוות ישנים נוירונים ([תאי עצב](#)), יחידות עיבוד קטנות בעלות מספר קלטים ומוצא אחד אשר משקלות באופן שונה את הכניסות באופן נלמד (סינפסות). התהlik שמתרחש ברשתות נוירונים אמייתות במוח שונה ורחוק מהותית ממה שמתתרחש ביום ברשתות נוירונים מלאכותיות ויש להתייחס לאנלוגיה בזיהירות.

נסתכל על רשתות נוירונים כאל פונקציה $f: \Omega \rightarrow X$. באמצעות Gradient Descent נמצא את המינימום של פונקציית השגיאה בין חיזוי הרשת לבין התיאוג האמתי של כל דוגמה.

בהתיחס לאיור 1 – תיאור סכמטי של תהליך הלימוד, שלב בחירת המודל נקבע ע"י בחירת **פונקציית השגיאה ובחירת מבנה (ארכיטקטורת)** הרשות. שלב אימון המודל נעשה ע"י **Gradient Descent** כדלקמן :

אלגוריתם האימון של רשת נוירוניים

קלט : סדרת דוגמאות מתיוגות לאימון $\{x_i, y_i\}_{i=1}^n$, פרמטר קצב הלימוד $0 < \eta$, מספר חזרות מקסימלי על סדרת הלימוד k .

פלט : פונקציה f המבוצעת באמצעות אוסף הפרמטרים w
שלבי הלימוד :

1. **אתחל** את הוקטור w^0 .
2. **רוץ** על דוגמאות סדרת האימון $i = 1, 2, 3, \dots$:
 - 2.1. קיבל את הוקטור x_i והתיוג y_i .
 - 2.2. קבע את קצב הלימוד η_t .
 - 2.3. חשב את שגיאת המודל עם וקטור המשקלים הנוכחי w^t : $f(x, y, w^t)$.
 - 2.4. חשב את וקטור הגרדיינט ($\nabla_w f(x, y, w)$) : $.g(x, y, w) = \nabla_w f(x, y, w)$.
 - 2.5. עדכן את וקטור המשקלות w^{t+1} : $w^{t+1} = w^t - \eta_t g(x, y, w)$
3. עברו שוב על סדרת הלימוד (שלב 2) עד שמקיים אחד משני תנאי העצירה הבאים :
 - 3.1. הושגה שגיאה נמוכה מספיק על סט המבחן.
 - 3.2. הושלמו k חזרות על סדרת הלימוד (epochs).
4. החזר את אוסף הפרמטרים w .

האלגוריתם לעיל מתאר את פועלות רשת הנוירונים באופן כללי. במהלך המעבדה נראה מספר מקרים פרטיים מפורטים שיתנו לכם טעימה מעולם הרשות.

סיווג באמצעות שגיאה ריבועית ונוירון בודד - ADALINE

כאמור, כל הרשותהן מקרים פרטיים של התהליך שמתואר לעיל. בעת נציג את המודל הפשטוט ביוור. בספרות האלגוריתם ידוע בשם **Widrow-Hoff**, **ADALINE** (Adaptive Linear Neuron) או **LMS(Least Mean Squares)**

פונקציית השגיאה שנרצה למוצר היא שגיאת הסיווג האמיתית או המקורבת ע"י חישוב השגיאה על סט האימון :

$$Err(f) = \frac{1}{n} \sum_{i=1}^n l(f(x_i), y_i)$$

$$\text{כasher } \ell(f(x_i), y_i) = \begin{cases} 0 & f(x_i) = y_i \\ 1 & f(x_i) \neq y_i \end{cases} : \text{(zero-one loss (נקראת השגיאה (neglect of loss))}$$

מה הבעיה בשימוש בפונקציית שגיאה זו?

שגיאה מסווג זה אינה גזירה! לכן לא יוכל להשתמש ב-Gradient Descent הפתרון: שימוש בפונקציית שגיאה חליפית (Surrogate Loss Function). לעומת זאת, שגיאה מזווערת יכולה לעזור שגיאת ה-zero-one.

פונקציית השגיאה שנבחר היא שגיאה ריבועית:

$$Err(f) = \frac{1}{n} \sum_{i=1}^n (f(x_i) - y_i)^2$$

כasher אנחנו מניחים ש- $y_i \in \{-1, 1\}$.

ארכיטקטורת הרשת תהיה פונקציה אפינית (שליעיתים מכונה, באופן מבלבל, שכבה ליניארית).

ליניארית= כפל במטריצה, אפינית= כפל במטריצה והוספת וקטור):

$$\sum_{d=1}^D w_d x_i^d + b$$

הערה על סימונים: x_i = הדוגמה ה- i -בסט האימון, x_i^d = המאפיין ה- d -של הדוגמה ה- i -בסט האימון. w הינו וקטור המשקלות של הפונקציה הליניארית ו- b הינו פרמטר ההטיה (bias).

צורת רישום חלופית: ניתן להוסיף איבר נוסף לקלט $1 = x_{d+1}$ ולסמן $w_{D+1} = b$, המאפשר

שימוש בכתב מקוצר:

$$f(x_i) = w^T x_i = \sum_{d=1}^{D+1} w_d x_i^d$$

כלומר, ע"י הגדלת גודל הקלט ב-1 והגדלת מספר קבוע בו (המספר אחד), והגדלת וקטור המשקלות ב-1 והגדלת איבר זה כ- b , הצלחנו להפוך את הביטוי המקורי $w^T x + b$ לביטוי הכלול אך ורק מכפלה יחידה $w^T x$. שימו לב לשינויו הרישום לא משנה את העובדה שהפונקציה אפינית.

לאיזו משפחה שייך המשתנה (x_i, f) ? \mathbb{R} ? \mathbb{N} ?

$f(x_i)$ הוא סכום ממושקל של משתנים רציפים, שכן גם הוא רציף. איך בכל זאת נקבע את תיוג המחלקה מתוך מוצא רציף?

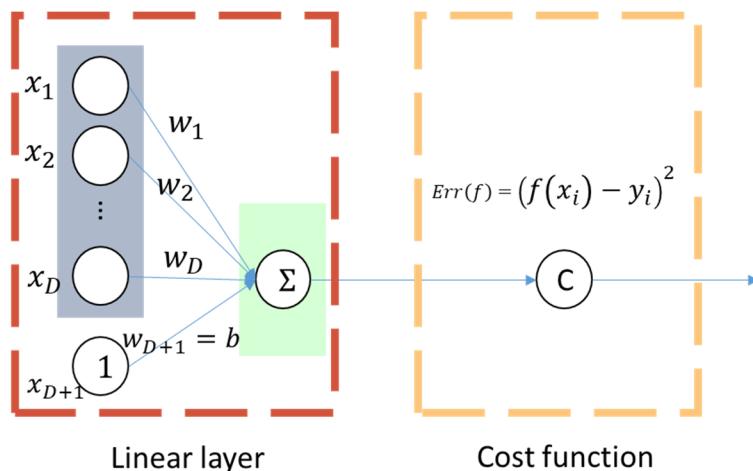
כאשר נבצע הסקה באמצעות המודל (כלומר נשאל על דוגמה חדשה מהי המחלקה שלה) نتيיג אותה באמצעות פונקציית Hard Limiter (הידועה גם בשם פונקציית מדרגה ו-heaviside), שמותאמת לסיוג בינארי בין שתי מחלקות:

$$\hat{y} = \varphi_{HL}(w^T x) = \begin{cases} -1, & w^T x < 0 \\ +1, & w^T x \geq 0 \end{cases}$$

נתבונן בפונקציה $x \mapsto g(x) = w^T x$. המשווה $g(x) = 0$ מגדירה משטח החלטה המפריד בין שתי מחלקות: $g(x) \geq 0$ ו- $g(x) < 0$, ובהתאם לכך נקבע הסיווג שננסמו \hat{y} .

כלומר משטח ההחלטה הינו על-מישור (על-מישור=מישור ביוטר מ-2 מימדים). כאשר $x \in \mathbb{R}^2$ אנחנו מקבלים קו ליניארי.

ניתן לתאר סכמתית את המודל באופן הבא:



איור 9 – סכמה של מודל ADALINE

1. אימון באמצעות Gradient Descent

כאמור, נעדכן באופן איטרטיבי את וקטור המשקלות במטרה להקטין את שגיאות הסיווג על סדרת הדוגמאות. וקטור המשקלות באיטרציה t יסומן w^t . על מנת לעדכן את הרשת יש לחשב את הגרדיינט של המודל:

$$\frac{\partial f}{\partial w} = \frac{\partial (w^T x - y)^2}{\partial w} = 2x(w^T x - y) \propto x(w^T x - y)$$

אלגוריתם אימון ADALINE

קלט: סדרת דוגמאות מתואימות לאימון $\{x_i, y_i\}_{i=1}^n$, פרמטר קצב ההתקנסות $0 < \eta_0$, מספר חזרות מקסימלי על סדרת הלימוד k .

פלט: וקטור פרמטרים $w^{t_{end}} = w^k = (w_1, w_2, \dots, w_{D+1})$.
שלבי הלימוד:

1. **אתחול**: אתחל את הווקטור w^0 בערך כלשהו.

2. **רוץ על דוגמאות סדרת האימון** $i = 1, 2, 3, \dots, n$:

2.1. קיבל את הווקטור $w^{t+1} \in \mathbb{R}^{D+1}$ והתיאוג $y \in \{-1, +1\}$ של הדוגמא $-i$.

2.2. חשב את יציאת הרשת עם וקטור המשקלים הנוכחי w^t : $\hat{y} = w^{t^T} x$.

2.3. קבע את קצב הלימוד: (עבור השימוש הבסיסי, $m = 1$)

$$\eta_t = \eta_0/m$$

2.4. עדכן את וקטור המשקלים w^{t+1} (שהוא וקטור המשקלים אשר התקבל לאחר איטרציה t):

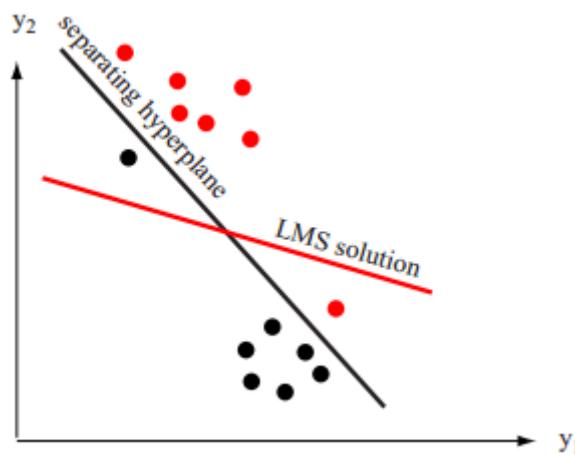
$$w^{t+1} = w^t - \eta_t x (\hat{y} - y) = w^t - \eta_t x (w^{t^T} x - y)$$

3. עבור שוב על סדרת הלימוד (שלב 2) עד שהוישלמו k חזרות על סדרת הלימוד (epochs).

4. החזר את w^k

הערות:

- האלגוריתם לעיל הוא מקרה פרטי של האלגוריתם שראינו קודם לאימון רשותות.
- במקרה הפחות הניל היה ניתן לפטור את מערכת המשוואות ע"י נוסחה סגורה במקומות התהיליך האיטרטיבי. החיסרונו של שימוש בנוסחה סגורה היא שלא ניתן להשתמש בה כאשר מספר הדוגמאות שלנו גדול מאוד.
- האלגוריתם לאו דווקא יתכנס לפתרון שנותן סיוג מושלם. לדוגמה:



אייר 10 – דוגמה לפתרון של LMS (ADALINE) שלא נותן סיוג מושלם, לעומת פתרון של מודל אחר שכן נותן סיוג מושלם

סימולציה - בעית סיווג האירוסים

דוגמה מפורסמת של בעיה שאotta ניתן לפתור באמצעות אלגוריתם לומד, הינה בעית [סיווג מיני אירוסים](#) עפ"י אורך ורוחב עלי הכותרת ועלי הגבע.

על מנת לפשט את הבעיה, נמש את אלגוריתם ADALINE על סמך שני מאפיינים (אורך ורוחב עלי הגבע) ושני מיני אירוסים (Setosa ו-Veriscolor.).

פתחו ב Matlab את הקובץ `adaline_iris.m`. (כמייל בקובץ `iris_adaline.m`).

*מומלץ למשח חלק זה לפני המשך הקריאה של חומר ההכנה.

זיכרון – התפלגות ברנולי

זהרי התפלגות של הטלת מטבע. ככלומר בהינתן פרמטר p , פונקציית פילוג ההסתברות:

$$P(x) = \begin{cases} p, & x = 1 \\ 1 - p, & x = 0 \end{cases} = p^x(1 - p)^{1-x}$$

גרסיה לוגיסטיבית

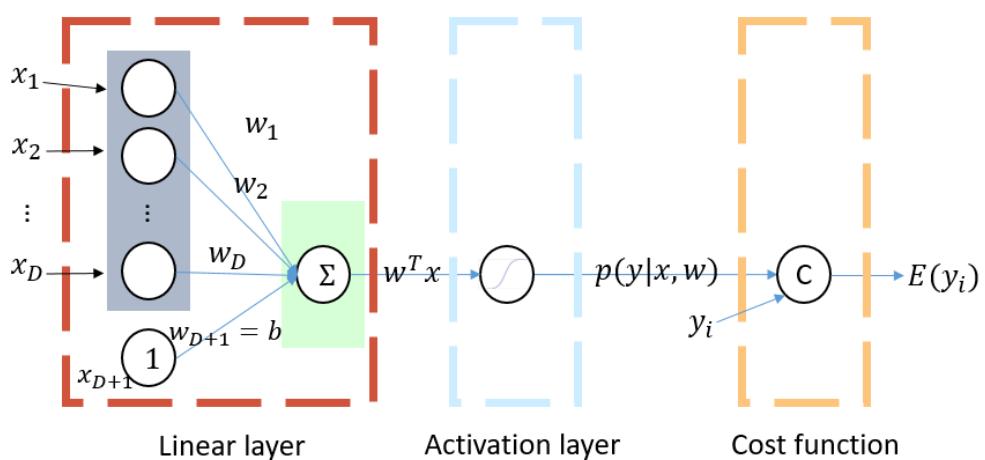
שימוש לב שבניו הראשו במעבדה תדרשו למשח גרסיה לוגיסטיבית בעצמכם.

גרסיה לוגיסטיבית הינו אלגוריתם סיווג ביןארי שפותח בשנת 1958 ע"י [David Cox](#).

אנחנו נתבונן על בעיה זו בעוד מקרה פרטי של רשת נוירונים. השוני העיקרי העיקרי, הוא שמודל זה מנשה לממד את ההסתברות של דוגמה להיות מחלקה מסוימת, ולא קביעה חד משמעית (hard) שהיא שייכת למחלקה.

ניתן להסתכל על מודל זה כמורכב משלוש שכבות: 1. שכבה ליניארית (אפינית) 2. שכבת אקטיבציה (שכבה לא-LINEARITY) 3. פונקציית מחיר (Cross entropy).

גרפית, נשרטט את המודל כך:



איור 11 – איור סכמטי של מודל לוגיסטי

כאמור, כעת מוצא המודל הינו $p(y|x, w)$.

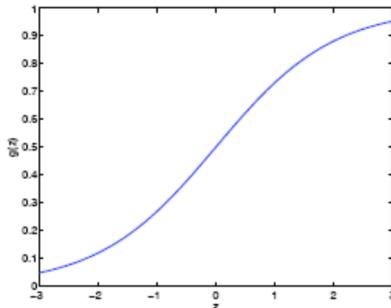
משיגים את הייצוג ההסתברותי ע"י שימוש בפונקציית הסיגמוואיד:

$$p(y|x, w) = \text{Ber}(y|\sigma(w^T x))$$

כאשר הסיגמוואיד מוגדר כך:

$$\sigma(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{1 + e^x}$$

וגרף נראה כך:



איור 12 – גраф של פונקציית הסיגמוואיד

כלומר, בהינתן וקטור משקלות w , ודוגמה x , הסיכוי של y_i להיות מחלוקת מספר 1 מתפלג לפי ברנולי עם פרמטר $\sigma(w^T x) = p$.

* שימושו לב שփונקציה מוציאה ערכים בין 0 ל-1, וכך מפונקציית פילוג הסתברות.
בצורה מפורשת, הביטוי אותו מחשבים הינו:

$$P(y_i|x_i, w) = \begin{cases} \pi_{i0} \triangleq 1 - \frac{1}{1 + e^{-w^T x}}, & y_i = 0 \\ \pi_{i1} \triangleq \frac{1}{1 + e^{-w^T x}}, & y_i = 1 \end{cases}$$

כלומר, π_{i0} היא ההסתברות עבורה y_i קיבל ערך 0 ו- π_{i1} היא ההסתברות עבורה y_i קיבל ערך 1.

כאמור, נרצה למזער את ההסתברות לשגיאה על פני כל הדוגמאות:

$$P(Y|X, w) \stackrel{i.i.d.}{=} \prod_{i=1}^n \text{Ber}(y_i|\sigma(x_i w)) \triangleq \prod_{i=1}^n \pi_{i0}^{1_{0}(y_i)} \pi_{i1}^{1_{1}(y_i)}$$

כאשר השתמשנו בהנחה ה-i.i.d. של הנתונים (התפלגות משותפת = מכפלת התפלגויות) ובהגדרת התפלגות ברנולי.
כמו כן,

$$1_{0}(y_i) = \begin{cases} 1, & y_i = 0 \\ 0, & \text{else} \end{cases} = 1 - y_i, \quad 1_{1}(y_i) = \begin{cases} 1, & y_i = 1 \\ 0, & \text{else} \end{cases} = y_i$$

מיקסום ההסתברות הניל שקול למזעור מינוס הלוג של אותו הביטוי:

$$NLL(Y, X, W) = - \sum_{i=1}^n 1_{0}(y_i) \log \pi_{i0} + 1_{1}(y_i) \log \pi_{i1} \left(= - \sum_{i=1}^n (1 - y_i) \log \pi_{i0} + y_i \log \pi_{i1} \right)$$

hbityoi hn'il n'kra minos log shg'at ha'stbarot (NLL = NLL) או hn' Entropy.

חישוב הגרדיינט

במקרה של רגרסיה לוגיסטי, הפונקציה אותה נרצה למצער הינה:

$$\begin{aligned} f(w, x, y) &= NLL(Y, X, W) = - \sum_{i=1}^n (1 - y_i) \log \pi_{i0} + y_i \log \pi_{i1} \\ &= - \sum_{i=1}^n (1 - y_i) \log(1 - \pi_{i1}) + y_i \log(\pi_{i1}) \end{aligned}$$

מהי הנגזרת של hbityoi hn'il?

כדי לחשב hbityoi mpofresh, nsh'tmesh b'k'l ha'shreret, casher ng'dir $x = w^T z$

$$g(w, x_i, y_i) = \frac{\partial f}{\partial w} = \frac{\partial f}{\partial \pi_{i1}} \frac{\partial \pi_{i1}}{\partial z} \frac{\partial z}{\partial w}$$

cut n'reshom at hbityoim b'zora mpofreshet:

$$\frac{\partial f}{\partial \pi_{i1}} = \frac{(1 - y_i)}{1 - \pi_{i1}} - \frac{y_i}{\pi_{i1}}$$

$$\frac{\partial \sigma}{\partial z} = \sigma(z)(1 - \sigma(z))$$

$$\frac{\partial z}{\partial w} = x_i$$

hochhat hbityoi hn'il v'hshlihi hn' sh'elot m'sefar 3 v-4 batr'gili ha'cnah.

שםו לב כי ב-(3) ביצעו גזירה מטריצית השיכת לחדו"א מטריצית. כל הזרה המטריציים מאד דומים לאלו של סקלרים. במעבדה זו לא ניתן להוכיחם. המעניינים בפיתוח המלא, יכולים למצאו הסבר תמציתי על הנושא ניתן למצוא [כאן](#).

אלגוריתם אימון רגסיה לוגיסטי

קלט: סדרת דוגמאות מותיוגות לאימון $\{x_k, y_k\}_{k=1}^n$, פרמטר $0 < \alpha$, מספר חוזרות מקסימלי על סדרת הלימוד m .

פלט: וקטור פרמטרים $w = (w_1, w_2, \dots, w_d, w_{d+1})$
שלבי הלימוד:

1. אתחל את הווקטור $w^0 = (1, 1, \dots, 1)$ בערך כלשהו (למשל,

$i = 1, 2, 3, \dots, n$,

2.1. קיבל את הווקטור x_i והתיוג $y_i \in \{0, 1\}$.

2.2. קבע את קצב הלימוד η .

2.3. חשב את שגיאת המודל עם וקטור המשקלים הנוכחי w^t :

2.4. חשב את הווקטור (w, y, x) .

2.5. עדכן את w^{t+1} (שהוא וקטור המשקלות אשר התקבל לאחר איטרציה t):

$$w^{t+1} = w^t - \alpha g(w, y, x)$$

3. עבור שוב על סדרת הלימוד (שלב 2) עד שמתקיים אחד משני תנאי העצירה הבאים:

3.1. אין יותר עדכון של w - האלגוריתם מנבא נכון את התיאוג של כל הדוגמאות.

3.2. הושלמו m חוזרות על סדרת הלימוד (epochs).

הערות:

- גם אלגוריתם זה הוא מקרה פרטי של האלגוריתם שראינו קודם לאימון רשותות.
- הערכים של y הפעם הם $\{0, 1\}$ במקום $\{-1, 1\}$. בפועל כל מספר מייצג מחלוקת (לדוגמה: סוג דג או סוג של פרח).

רגסיה לוגיסטית כמודל שכבות

כעת ננסה להכליל את שלבים 2.2 ו-2.3 למודל שכבות כללי תוך הסתכלות על רגסיה לוגיסטית במקרה בווחן.

רשתות נירונים מאימים באמצעות שתי פונקציות רקורסיביות:

1. פונקציית מעבר קדמית (שלב 2.3).

2. פונקציית מעבר אחוריית (שלב 2.4).

פונקציית המעבר **קדמית** של המודל מוגדר כהרכבה של השכבות:

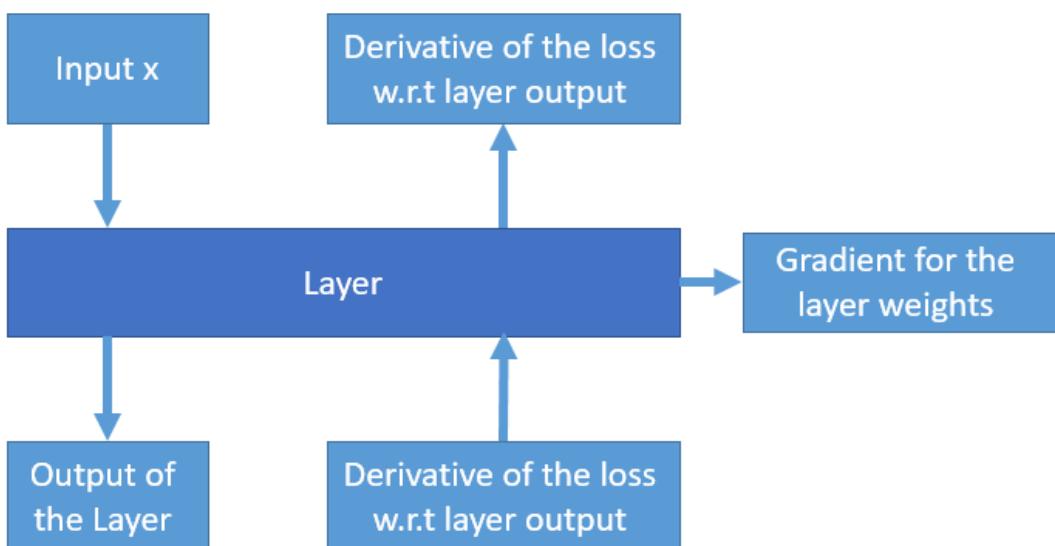
$$f(x, y) = C(\sigma(L(x)))$$

כאשר C היא פונקציית המחיר, σ היא הסיגמוואיד ו- L היא הפונקציה הליינארית. פונקציית המעבר האחורי היא למעשה חישוב הנגורות לכל שכבה. פעם לפि הקלט של השכבה, ופעם לפি הפרמטרים של השכבה אם קיימים:

$$g(w, x_i, y_i) = \frac{\partial f}{\partial w} = \frac{\partial f}{\partial \sigma} \frac{\partial \sigma}{\partial z} \frac{\partial z}{\partial w}$$

לאחר המעבר הקדמי והאחורי, ניתן לעדכן את משקלות (פרמטרי) המודל לפי כל מודל מתקדם אחר או כל מודל מתקדם אחר.

נוכל להסתכל על כל שכבה כיחידה מודולרית ועצמאית:



איור 13 – שכבה כללית בראשת ניירוניים. כניסה ויציאות

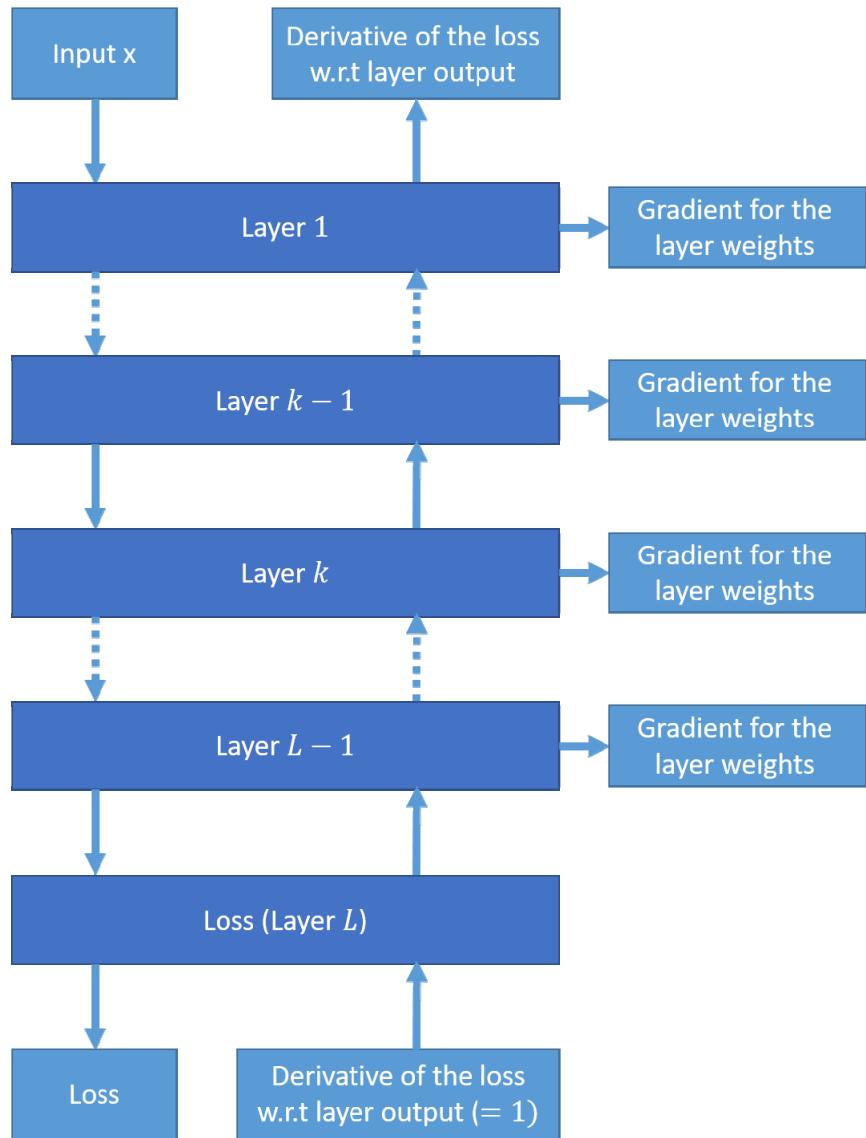
על מנת שנוכל להשתמש בה בלמידה הרשת, עליה למשמש שלוש פונקציות:

1. פונקציית מעבר קדמית.
2. נגזרת לפי פרמטרים.
3. נגזרת לפי הכניסה.

רשמו בצורה מפורשת את שלוש הפונקציות של השכבות ברגסיה לוגיסטיית עבור המקרה הסקלירי:

מעבר קדמי	נגזרת לפי הכניסה	נגזרת לפי פרמטרים	שכבה ליינארית
sigmoid	-	-	linear
NLL	-	-	

לאחר שהיחסנו (מיימנו) את שלוש הפונקציות, ניתן לחבר את השכבות יחדיו ולאמן את כל המודל באמצעות אימון זו מכונה בשם backpropagation. למעשה, מדובר באלגוריתם מינימום מושג, מדויק, יותר מאשר הפעלת כלל השרשרת.



איור 14 – רשת נוירונים כמודול רב שכבותי

בעיית סיווג מתוך מחלקות רבות

עד כה דנו בבעיות סיווג ביןארית- בהינתן נקודה במרחב עלינו להחליט לאיזו מחלוקת מבין שתיים נתונות הנקודה שייכת. במקרה כזה, בראשת הנוירונים שאנו בונים מספיקה יציאה יחידה אשר תקבע את השתיכות הכניסה (1 או 0).

כאשר אנו צריכים לקבוע שיווק לאחת מ- N מחלקות, וקטור הסיווג הנוכחי יהיה וקטור בגודל N ויכיל אפסים בכל המיקומים פרט לאחד, באינדקס המציין את המחלוקת אליה משתייכת הנקודה הרלוונטי. נרצה שモצאה הראשית יהיה וקטור בגודל N המכיל עבור כל אינדקס את ההסתברות של

השתייכות הcnisa לחלוקת באינדקס זה. כלומר, וקטור של N ערכים בין 0 ל-1, כאשר סכום הערכים הוא 1, והערך הגבוה ביותר יתאים לחלוקת אשר ההשתייכות אליה תהיה בעלת הסבירות הגבוהה ביותר.

על מנת להגיע לモאָץ כנדרש, נשתמש בפונקציית אקטיבציה הנקראת softmax. נסמן את מוצא השכבה הקודמת ל-softmax ע"י o .

$$\text{softmax}(o_i) = \frac{e^{o_i}}{\sum_{k=1}^N e^{o_k}}$$

שימוש לב Ci עבור כלחלוקת נקבע כנדרש, ערך בין 0 ל-1 כאשר סכום על הערכים על המחלוקות הוא 1.

גם עבור פונקציה זו, נחשב את הגרדייאנט עבור ה- softmax .

$p_i = \text{softmax}(o_i) \cdot \frac{\partial \text{softmax}(o)}{\partial x_i}$. נסמן נחשב נפריד למקרים :

$$\frac{\partial \text{softmax}(o_i)}{\partial o_i}, \frac{\partial \text{softmax}(o_{j \neq i})}{\partial o_i}$$

בתרגיל ההכנה תראו כי מתקיים :

$$\frac{\partial p_i}{\partial o_j} = \begin{cases} p_i(1 - p_i), & i = j \\ -p_i p_j, & i \neq j \end{cases} = p_i(\delta_{ij} - p_j) \quad \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

על מנת לפשט את הביטוי, נגזר את שכבת ה-softmax יחד עם שכבת ה- NLL הסופית ברשות. עבור בחירה מתוך N מחלוקות, פונקציית ה- NLL נראה כך :

$$\text{NLL}(Y, X, W) = - \sum_{i=1}^N y_i \log p_i$$

$$\frac{\partial \text{NLL}}{\partial p_i} = - \frac{y_i}{p_i}$$

נגזר את הפונקציה NLL ביחס למשתנה הcnisa של ה-softmax : o_j

$$\frac{\partial \text{NLL}}{\partial o_j} = - \sum_{i=1}^N y_i \frac{\partial \log(p_i)}{\partial o_j} = - \sum_{i=1}^N \frac{y_i}{p_i} \frac{\partial p_i}{\partial o_j} = - \sum_{i=1}^N \frac{y_i}{p_i} p_i (\delta_{ij} - p_j) = - \sum_{i=1}^N y_i (\delta_{ij} - p_j)$$

$$= -y_j + p_j \underbrace{\sum_{i=1}^N y_i}_{=1} = p_j - y_j$$

קריאה נוספת

[1] *Pattern Classification* (2nd ed.), Richard O. Duda, Peter E. Hart and David G. Stork (John Wiley and Sons, 2001)

[2] *Deep Learning*, Ian Goodfellow and Yoshua Bengio and Aaron Courville (MIT Press, 2016)

מפגש ראשון

שאלות הכנה

1. עברו כל אחת מן הביעות הבאות קבעו: האם מדובר בבעיתת קלסיפיקציה או רגרסיה, הצעו מרחב דוגמאות ומרחב תיוג.

- א. זיהוי כתוב יד מהתמונה
 - ב. מסנן דואר זבל (Spam)
 - ג. זיהוי דבר מקטע אודיו
 - ד. חיזוי שער מניה לאחר מספר ימים
 - ה. חיזוי משך נסעה בין שני יעדים
- הצעו עוד שתי דוגמאות לביעות משלכם ונתחו אותם באותו האופן.

– 2. עברו כל אחת מהשכבות הבאות (כל אחת בנפרד), פתחו את שלוש הפונקציות (forward, הפונקציה עצמה, ושני ה-backward-גזרה של הפונקציה לפי הקלט ולפי הפרמטר) :

1. $f(x, a) = \log(ax)$
2. $MSE(x, y) = (x - y)^2$
3. $f(x) = \max(0, x)$
4. $f(x, a, b) = \sqrt{ax + b}$

* ניתן להניח שהפונקציות והמשתנים סקלריים.

3. עברו הפונקציה המטריצית הבאה : $f: \mathbb{R}^3 \rightarrow \mathbb{R}^2$

$$f(X) = A^T X = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$$

כאשר $A \in \mathbb{R}^{3 \times 2}$ ו- $X \in \mathbb{R}^3$.

פתחו את פונקציות ה-backward של השכבה.

שימוש לב שהגרדיינט של f נתון ע"י :

$$\nabla_X f(X) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} \end{bmatrix}$$

הדרך: כדי לחשב את ה-backward של השכבה ביחס לפרמטרים נגדיר את השכבה בצורה הבא :

$$f(X) = \begin{bmatrix} f_1(X) \\ f_2(X) \end{bmatrix} = \begin{bmatrix} A_1^T X \\ A_2^T X \end{bmatrix}$$

כאשר :

$$A_1 = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix}, A_2 = \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \end{bmatrix}$$

הנגזרת של f לפי A מוגדרת כך :

$$\nabla_A f = \begin{bmatrix} \nabla_{A_1} f_1 \\ \nabla_{A_2} f_2 \end{bmatrix}$$

4. יש להראות כי הנגזרת של הפונקציה $(z)\sigma$, נתון ע"י הביטוי :

$$\frac{\partial \sigma}{\partial z} = \sigma(z)(1 - \sigma(z))$$

Gradient descent .5

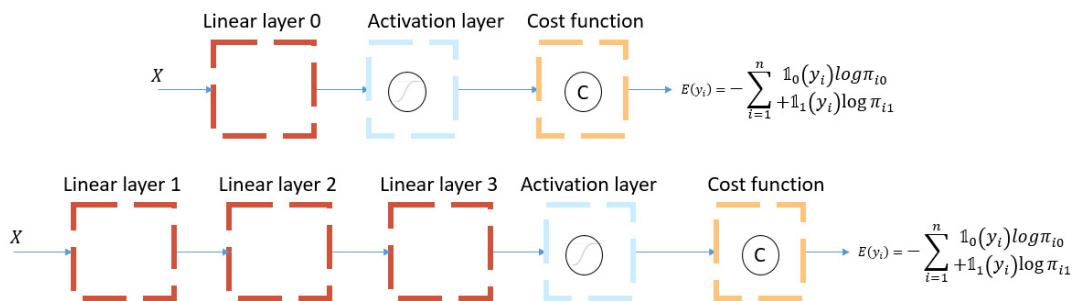
בහינתן הפונקציה $f(x) = x^2$ ונקודת התחלה $x_0 = -1$, יש להציגם שלוש איטרציות של *gradient descent*

א. עבור גודל צעד $\eta = 0.01$.

ב. עבור גודל צעד $\eta = 2$.

מהי חשיבותו של בחירת גודל הצעד בתהליכי הלמידה ?

6. יש להראות כי הרשנות הבאות שකולות :



7. הגישו את הקוד ל- ADALINE עבור סיווג האירוסים.

8. הגישו את הطلبת המלאה מלוגיסטייה לוגיסטיית כמודל שכבות בה יש למלא את הפונקציות של כל שכבה במודול.

9. עבור פונקציית softmax, פתרו את הביטויים לנגזרות והראו שמתקיים :

$$\frac{\partial p_i}{\partial o_j} = \begin{cases} p_i(1 - p_i), & i = j \\ -p_i p_j, & i \neq j \end{cases} = p_i(\delta_{ij} - p_j) \quad \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

מפגש ראשון - מהלך הניסוי

בניסוי זה נמשך רגסיה לוגיסטיבית עבור בעיית סיוג האירוסים בדומה לנעשה בתרגיל ההכנה. לאחר מכן ננסה ברשותנו עיקומות הכוללות מספר גדול יותר של שכבות לסיוג ספרות מותוק תמונה. לבסוף נכיר את ה-toolbox המובנה של MATLAB לבניה ואמון של רשות.

הנחיות

שאלות הדוח'ח המשכם מופיעות לאורך הניסוי. יש לענות עליהם בכיתה, במהלך ביצוע הניסוי: בכל פעם שתצטרכו לבצע משימה כלשהי במהלך הניסוי, המשימה תופיע ליד סימן קבוע. לדוגמה: ♦ הריצו את הקובץ `deepLearningIsFun.m` (דוגמא של סימון משימה בקובץ במהלך הניסוי).

השאירו את קבצי הקוד בתיקיות כפי שקיבלתם אותם (אין צורך להוציא את כולם לאותה תיקיה). **בנוסף, עליכם לצרף לדוח'ח המשכם את כל קבצי ה-MATLAB (קוד, תוצאות וגרפים) שייצרתם במהלך הניסוי.**

דרישות מחשוב:

MATLAB 2018a –

- Neural Network Toolbox
- Parallel Computing Toolbox
- Statistics and Machine Learning Toolbox
- AlexNet pre-trained network

חלק א' – רגרסיה לוגיסטיבית

בשלב הראשון בניסוי, נמשך רגרסיה לוגיסטיבית עבור בעית סיווג האירוסים.

נזכיר, כי רשותות נוירונים מאמנים באמצעות שתי פונקציות רקורסיביות :

1. פונקציית מעבר קדמית (שלב 2.3 באלגוריתם).

2. פונקציית מעבר אחורי (שלב 2.4 באלגוריתם).

ניעזר בטבלה ש חישבנו | בתרגיל הינה מס' 8 על מנת למשר רשות בשלבים.

בזומה לתרגיל הינה, מצורפים קטעי קוד MATLAB חלקים אשר ממשים מודולים המרכיבים

מערכת היררכית של רגרסיה לוגיסטיבית :

- `logistic_regression.m`

- o `forward functions:`

- `affine_forward.m`

- `nll_forward.m`

- `sigmoid_forward.m`

- o `backward functions:`

- `affine_backward.m`

- `nll_backward.m`

- `sigmoid_backward.m`

1. עבור רגרסיה לוגיסטיבית כמודל שכבות- בהינתן וקטור כניסה לרשות $\mathbb{R}^d \in x$ ווקטור מוצא

$\mathbb{R} \in u$, מהו הממד של x וקטור המשקלות הכלול בתוכו את רכיב ה-*bias*?

❖ הסתכלו בקבצי MATLAB הנתונים לכם ופתחו את התיקייה `lab1`.

2. הסתכלו על הפונקציות הנתונות בתיקייה `layers`. מהו סדר השימוש הנכון בפונקציות עבור

תהליך `forward` ועבור תהליך `backward`?

- o שימו לב כי כל אחת מהפונקציות מתאימה לפונקציה בטליה שהגשתם בתרגיל ההינה.

a. מהו הממד הצפוי לモץ כל אחת מהפונקציות הללו (עבור וקטור כניסה $\mathbb{R}^d \in x$)?

❖ השלימו את הקוד והריצו טסטים עבור הפונקציות `affine_forward`, `nll_forward`,

`sigmoid_forward`, `affine_backward`, `nll_backward`, `sigmoid_backward` וודאו כי אכן

מתאים הממדים להם צייפיתם בסעיף הקודם.

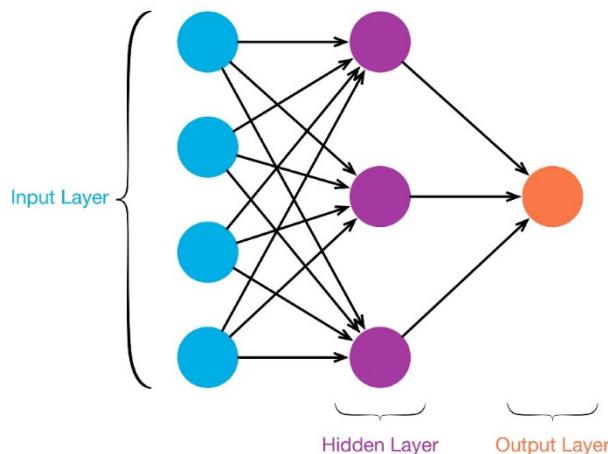
דגשים חשובים שצורך לקרוא לפני שמתחילה למש:

- כל פונקציית backward צריכה להחזיר את הגרדיאנט המלא הנוכחי, כלומר הגרדיאנט המחשב בצעד הנוכחי כפול הגרדיאנט שוחשב עד כה (לפי כלל השרשרת).
 - על הימימוש שלכם עבור כל פונקציה לתמוך בכניסה ייחידה (א' שהוא וקטורי עמודה) וגם בכניסה שהיא שרשור של מספר כניסה (א' שהוא מטריצה, שרשור של מספר וקטורי עמודה).
 - **השתמשו בקבצי הבדיקה המצורפים בתיקייה layers/tests:**
 - הריצו את הקובץ `.run_tests.m`
 - וראו כי הבדיקות עברו. בפלט ההערכתה תוכל לראות פירוט של הטיטים המורצים ואת הבדיקות אשר עברו או נכשלו.
 - במידה וקיים בדיקות שנכשלו, תקנו את הפונקציות הרלוונטיות לבדיקה וחזרו על הבדיקה.
- ❖ השלימו את הקוד עבור הפונקציה `logistic_regression`.
3. מהי הצורך שבה אתם מצפים שיהיה קו הפרדה של הרוגסיה הלוגיסטיבית?
- ❖ הריצו את `iris_logistic_regression`, עם אחד עם קצב התכניות 0.1 ופעם שנייה עם קצב התכניות 0.01. צפו בתהליכי ההתקנים. הוסיפו לדוח המסכם שלכם את פלטי הריצות.
4. ציינו את ההבדלים בין שתי ההצלחות שביצעתם והסבירו - מה גורם להבדלים אלו?

חלק ב' – זיהוי ספרות בתמונות (סיווג בעזרת רשת נוירוניים)

כעת נעבור לבעה יותר מורכבת: זיהוי ספרות כתוב יד. מאגר המידע בו נשתמש הוא MNIST, המכיל תמונות של ספרות, ותיוגים מתאימים של הספרה המתאימה לכל תמונה. כמובן, הקלט לרשת במקרה זה יהיה תמונה, והפלט הרצוי יהיה הקטגוריה המתאימה בספרה בתמונה (Classification).

❖ פתחו את הקובץ `MNISTapplyTwoLayerPerceptron` והוא ריצו את המקטע הראשון (`mnistDataPrep`). התבוננו בתמונות לדוגמה, ובחלוקת התוצאות. הרשת שאנו נבנה תהיה בעלת שתי שכבות. כמובן, כל שכבה מכילה הכפלת מטריצה (ליניארית), ואקטיבציה. בהתאם, לרשת תהיה שכבה נסתרת אחת של נוירונים. כמובן, הרשת תיראה כך (איולוסטרציה בלבד, הגדים אצלכם כמובן שונים):



שימוש לב שאות גודל השכבה הנסתרת ניתן לבחור כרצונו.

כל שכבה ליניארית באה כדי ביטוי עלי טור של חצים השחורים, והאקטיבציה מתבצעת בכל ניירון (עיגול סגול או כתום).

שכבת האקטיבציה הראשונה בה נשמש היא סיגמוד, בשכבה השנייה נשמש באקטיבציה מסווג softmax ופונקציית ההפסד היא Negative Log Likelihood.

1. רשמו את הביטויים המתמטיים עבור שלבי תהליך ה-forward של רשות צו. התיחסו לכל שלבי הבניינים (אחרי כל שכבה ליניארית/אקטיבציה/loss).

השתמשו בסימונים הבאים :

- a. x וקטור הכנסה לרשות
- b. y וקטור המוצא של הרשות
- c. w_1, w_2 המשקלות בשכבה הראשונה והשנייה בהתאם
- d. b_1, b_2 רכיב ה-bias בשכבה הראשונה והשנייה בהתאם
- e. $\sigma(x)$ פונקציית האקטיבציה סיגמוד
- f. $\text{softmax}(x)$ פונקציית האקטיבציה softmax
- g. \hat{y} פונקציית ההפסד

איך גודל השכבה הנסתרת משפיע על הביטויים המתמטיים?

כפי שראינו, כדי לאמן את המודל, יש לחשב את שלושת הפונקציות עבור כל שכבה. למעשה, כבר חישבנו כמעט את כל השכבות והגזרות הנחוצות למודל שבו שיצנו את הרגression הלוגיסטי.

נשים לב כי במקרה של MNIST, בנויגוד למשימה הקודמת אנו בוחרים מחלוקת אחת מתוך 10 מחלקות ולא מביצים סיווג בינארי. לכן, יש להתאים את פונקציית ה-TAN.

❖ הסתכלו בתיקייה המתאימה למשימה זו. פתרו את התיקייה layers-mnist, והשלימו את

הקוד עבור הפונקציות הבאות לפי הנוסחאות המוצגות בחומר הרקע למעבדה :

```

nll_forward_mnist.m   ○
softmax_forward.m    ○
nll_and_softmax_backward.m ○

```

גם כאן תקפים אוטם הציגים חלק א'. השתמשו בטסטים הניטנים لكم על מנת לבדוק את הפונקציות שכתבתם. שימושם לבן כל השרשרת פועל גם **באופן וקטורי**. לכן, שימושו לבן לממדיה ה嚮前传播 והשימוש ב-*** מול**.

בקטעי הקוד מצורפים لكم קבועים הממשים רשות זו. הקבועים מחולקים באופן הבא :

- applyTwoLayerPerceptronMNIST – מעתפת כללית אשר מכינה את מאגר המידע, שולחת אותו לאימון ולבסוף שולחת לוlidציה ומחשבת error.
- forward_pass – מבצעת מעבר קדמי על הרשות back-propagation – מבצעת back-propagation על הרשות
- trainTwoLayerPerceptron – מאמנת את הרשות שלנו ומחזירה את המשקولات לאחר האימון.
- validateTwoLayerPerceptron – מבצעת forward pass על סט הולידציה ומחזירה שגיאה.

- ❖ שלימנו את השורות הנחוצות בtrainTwoLayerPerceptron.
 - ❖ שלימנו את הקוד ב-forward_pass וbackward_pass על ידי השכבות שכתבתם בעצמכם בחלק א' ובחלק ב'.
 - ❖ הריצו את הtrainTwoLayerPerceptronMNIST. ודאו כי המודל מתכנס, וצרפו את התוצאות הסופיות והגרף המתkeletal.
2. מהו גודל השכבה הנסתורת כרגע בקוד? הסבירו כיצד גודל השכבה הנסתורת משפיע על המודל. התיחסו בתשובתכם לSkills המודול ושיקולי鄙视 (מקום, זמן, דיווק). (הציגו באירוע 5 ואירוע 6 לקבלת אינטואיציה)

חלק ג' – סיווג ספרות בעזרת Matlab Neural Network Toolbox

שימוש לב- חלק זה אינו תלוי בחולקים הקודמים במעבדה זו.icut, נבנה רשות זהה לרשות אשר בנינו בחלק ב' בעזרת כל רשות הנוירונים של MATLAB. מאגר המידע בו נשתמש הוא מאגר דומה מאוד ל-MNIST, המכיל תמונות ותיוגים שהם הספרות בתמונה.

- ❖ קראו את נספח א' – Matlab Layers – חלק א', והבינו את מטרתה של כל שכבה. בኒיגוד לחלק הקודם, האקטיבציה הראשונה בה משתמש היא ReLU, ולא סיגמוד. בחלק ההכנה למפגש הבא נלמד מה ההבדל ביניהם ומה היתרונות של ReLU. השימוש בשכבת ה-ReLU זהה לשימוש שעשינו בסיגמוד. האקטיבציה השנייה שששתמש בה תהיה softmax.

- ❖ בקובץ `mnistClassificationDL.m`, השלימו את השכבות הרצויות לפי המודל מהסעיף הקודם עם השינויים שצוינו בפסקה הקודמת.
- עליכם להחליף כל שורה המתארת שכבה בשכבה של מطلב המתאימה לשורה זו.
- רשימת השכבות הרלוונטיות לניסוי זה נמצאת בדף א' – Matlab Layers – חלק א'. שימוש לב פרמטרים של כל שכבה.
- הסבירו מה המשמעות של כל אחד מהפרמטרים המתקבלים כקלט לפונקציה `.mnistClassificationDL trainingOptions`

- ❖ הריצו את `mnist` וצפו בתהיליך ההתקנסות של הרשת. שימוש ב `mnistClassificationDL` ב MATLAB command window של `Base Learning Rate`:

Epoch	Iteration	Time Elapsed (seconds)	Mini-batch Loss	Validation Loss	Mini-batch Accuracy	Validation Accuracy	Base Learning Rate

הפרמטרים אשר יעניינו אותנו בעיקר בחלק זה הם `h-loss` ו-`h-accuracy`, של ה-`train` (לשום *C-Mini-batch*) ולמול `Validation` (*batch*). ניתן לראות את `h-loss` ו-`h-accuracy` של סט האימון בגרף ההתקנסות אשר מוצג בתהיליך הלמידה. שימוש לב שמותגים בכו מכווקו נתוני `h-validation`.

- ❖ הריצו את המודל במשך 10 epoch'ים עם גודל צעד 0.0001. הסתכלו על התקנסות `h-loss` לאורך האימון.
- ❖ הדיבקו את גרף ההתקנסות ואת הטבלה מה-`command accuracy`. רשמו את `h-loss` ו-`h-validation`. הסופיים של `h-train` ו-`h-validation`.
- מה הסיבה להבדלים בין `h-train loss` ובין `h-validation loss`? איזו תופעה מתרכחת כאן? איך לדעתכם ניתן לפתור את התופעה זו?

במפגש השני של המעבדה נראה מספר דרכי להתמודד עם תופעה זו ונדון בהן רבות. בסעיפים הבאים אתם יכולים להתעלם מטופעה זו בתשובותיכם.

- ❖ כתע, הריצו את המודל עם גודל צעד 0.1.
- ❖ הדיבקו את גרף ההתקנסות ואת הטבלה מה-`command accuracy`. רשמו את `h-loss` ו-`h-validation`. הסופיים של `h-train` ו-`h-validation`.
- מה קרה לגרף ההתקנסות? הסבירו מדוע.

❖ מצאו את גודל הצעד הגדול ביותר עבורו הרשות מצליחה להכנס תוך 10 epochs (דיוק של 3 ספרות אחרי הקודה בגודל הצעד), כלומר מצאו גודל צעד עבורו ניתן לראות מגמה מתאימה עבור ה-loss וה-accuracy.

4. השוו בין תהליכי ההתקנסות והתוצאות הסופיות עם גודל צעד 0.0001, 0.001. מה ההבדלים ביניהם? למה שינוי גודל הצעד הביא להבדלים אלו? התיחסו לנ吐ונים בטבלה ולצורת הגראף.

5. מה הייתם מצלפים שקרה עבור גודל צעד 0.00001? (מוזמנים להריץ ולבזוק את תשובתכם)

a. מהו גודל הצעד המתאים ביותר מבין שלושת האופציות ? 0.001, 0.0001, 0.00001?

האם גודל זה אידיאלי לאורך כל תהליכי ההתקנסות?

b. האם גודל זה יתאים לכל רשות נוירונים שנרצה לאמן? מדוע?

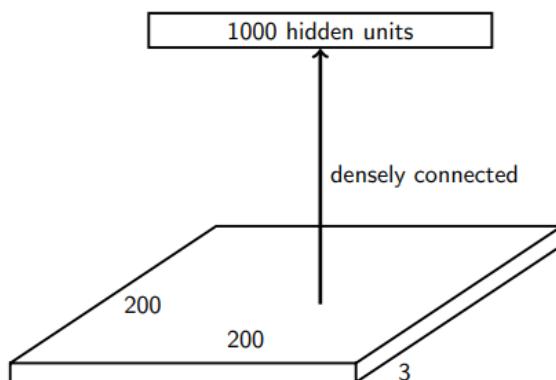
רקע למעבדה – מפגש שני

הקדמה

בחלק השני של הניסוי נכיר שכבות נוספות המהוות אבני בסיס בלמידה عمוקה. בפרט, נכיר שכבה הנקראת "שכבת קונבולוציה". לשכבה זו חשיבות גדולה בלמידה عمוקה והיא מאפשרת לתהום גדול لأن שהוא נמצא היום. כמו כן, נלמד על שכבות אי לינאריות, גישות אופטימיזציה מתקדמות, רגולרייזציה ושיקולים בתכנון רשתות.

מבוא

במעבדה הקודמת השתמשנו בתמונות מאוד קטנות – $1 \times 28 \times 28$. 1 מייצג את העובדה שמדובר בתמונות רמות אפור. בתמונה צבע יש 3 שכבות צבע. בתחומים רבים נרצה לעבוד עם תמונות יותר גדולות. למשל תמונות בגודל $200 \times 200 \times 3$. נציג שטמונות אלו קטנות מאוד בסטנדרטים של ימיינו. יש בוחן רק $k = 4k = 200 \times 200$ פיקסלים, לעומת זאת מצלמות פלאפון סטנדרטיות שמצילות תמונות עם מיליון פיקסלים. רשת נירונית שמקבלת תמונה כזו יכולה לעשות להירות כך:



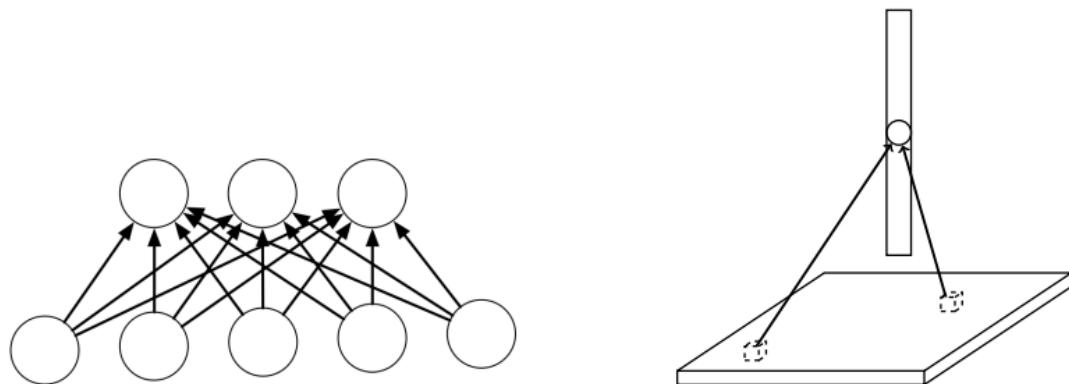
מה הביעה עם שכבה זו?

יש בה יותר מדי פרמטרים! לשכבה הראשונה (בלבד) יש $200 \times 200 \times 3 \times 1000 = 120 \text{ million}$. זו כמות גדולה מאוד של פרמטרים. הן מבחינת המוקם שהיא צריכה בזכרון (32 ביט לכל פרמטר) והן מבחינות זמן האימון שיידרש עד שכל הפרמטרים יותאמו לדוגמאות האימון. בעיה נוספת: מה יקרה אם נצלם את (כמעט) אותה התמונה רק בהיסט קטן ימין? שימו לב שבמקרה זה כל פיקסל יוכפל בפרמטר אחר! היינו רוצים לנצל תכוונה של תמונות טבעיות שהיא שווה לאובייקט לא תלולה במיקומו בתמונה. פרצוף בצד התמונה הוא פרצוף גם אם הוא נמצא במרכז התמונה. תכוונה זו נקראת אינוריאנטיות להזזה.

כעת נבנה בהדרגה את רשת הקונבולוציה, שפותרת את בעיות אלו.

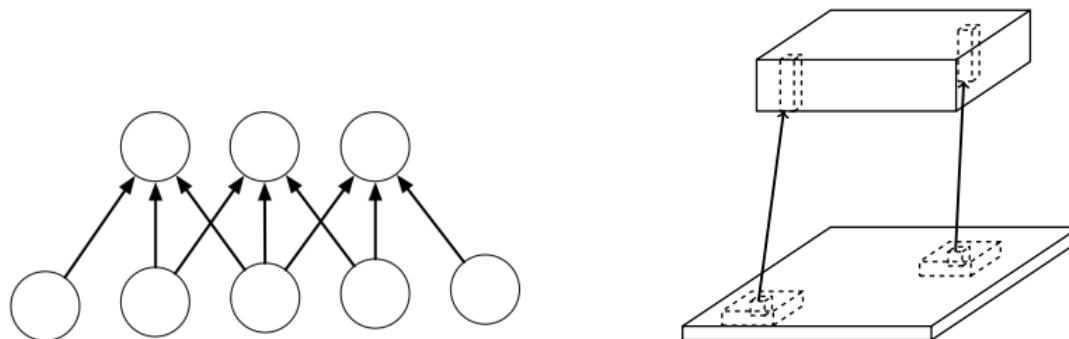
רשתות קוונטולזיה

סכמתית נתאר רשתות לינאריות כך :



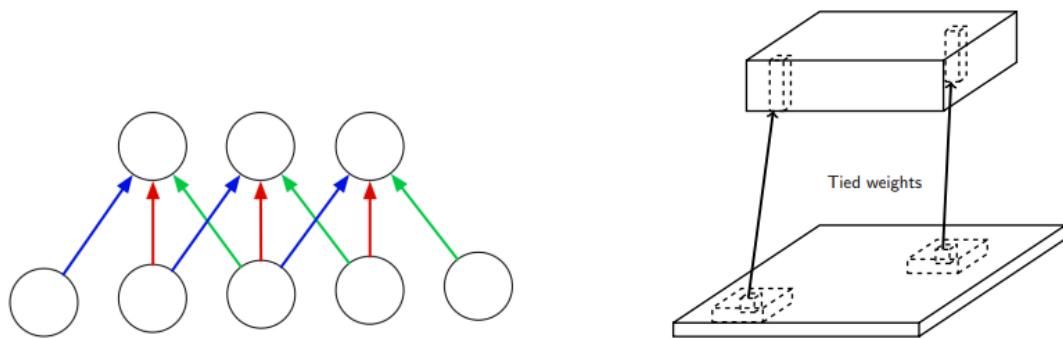
כל איבר בМОץ הוא מכפלה של כל הפיקסלים בתמונה בסט המשקولات שלו. ניתן גם לחשב על כל איבר בМОץ כקומבינציה לינארית של כל הפיקסלים בתמונה. שכבות לינאריות עלייתים מכונות גם שכבות **fully-connected**, כלומר בעלות חיבוריות מלאה.

כעת נשנה מעט את השכבה. נרצה שМОץ השכבה תהיה גם היא מטריצה תלת מימדית (גובה \times רוחב \times עומק). עומק במקרה של תמונה צבע=3). כמו כן, נדרש שכל פיקסל יושפע רק מסביבה קטנה בתמונה הכניסה. השכבה הניל תיראה כך :



שכבה כזו מכונה שכבת **locally connected**, או בעלת חיבוריות מקומית. שימוש לב שכל פיקסל בМОץ מייצג ע"י וקטור, בו כל איבר הוא קומבינציה לינארית **שונה** של האיברים בסביבתו (גובה, רוחב ועומק) בתמונה הכניסה.

כעת נעשו שינויים נוספים : המשקولات בכל המיקומים השונים יהיו שוים :



כלומר כל אזור בתמונה המוצא הוא תוצאה של מכפלות של אוטו משקלות עם אזוריים שונים בתמונה הכניסה. כל אוסף כזה של משקלות שמועבר על כל התמונה נקרא גרעין קונבולוציה או בקיצור גרעין.

פעולה זו היא בדיק פועלות הקונבולוציה שנלמד בקורס "מבוא לאותות ומערכות" רק בדו-מימד.³ מתמטית, הביטוי נראה כך:

$$(f * g)[n] = \sum_{i=0}^m \sum_{j=0}^k f[i][i+j]$$

דוגמה לkonvolוציה:

נראה כיצד נראה פועלות הקונבולוציה עבור תמונה בגודל 5×5 עם ערכים בינאריים.⁴ ערכי התמונה הם הערכים שופיעים בגודל עם רקע ירוק-צהוב. ערכי המשקלות מופיעות בקטן בצד אדום עם רקע צהוב. תוצר פועלות הקונבולוציה היא התמונה שופיע מימין בורוד. שימו לב שבאזור עם רקע צהוב אם נכפול את כל המספרים הגדולים במספרים הקטנים ונסכום, נקבל את הספרה 4 כפי שופיע בתמונה הימנית בפינה הימנית התחתונה.

1	1	1	0	0
0	1	1	1	0
0	0	1 _{x1}	1 _{x0}	1 _{x1}
0	0	1 _{x0}	1 _{x1}	0 _{x0}
0	1	1 _{x1}	0 _{x0}	0 _{x1}

Image

4	3	4
2	4	3
2	3	4

Convolved Feature

³ זו היא כמעט אותה פעולה. במספרות של עיבוד אותו נהוג להגדיר את הקונבולוציה באמצעות היפוך של אחת הפונקציות. הפעולה לה אנחנו מציגים כאן, לא ההיפוך, נקראת פועלות קרייס-קונולוציה. אנחנו ניצמד למוניה "קונבולוציה" מכיוון שהוא המונח המקובל בתחום.

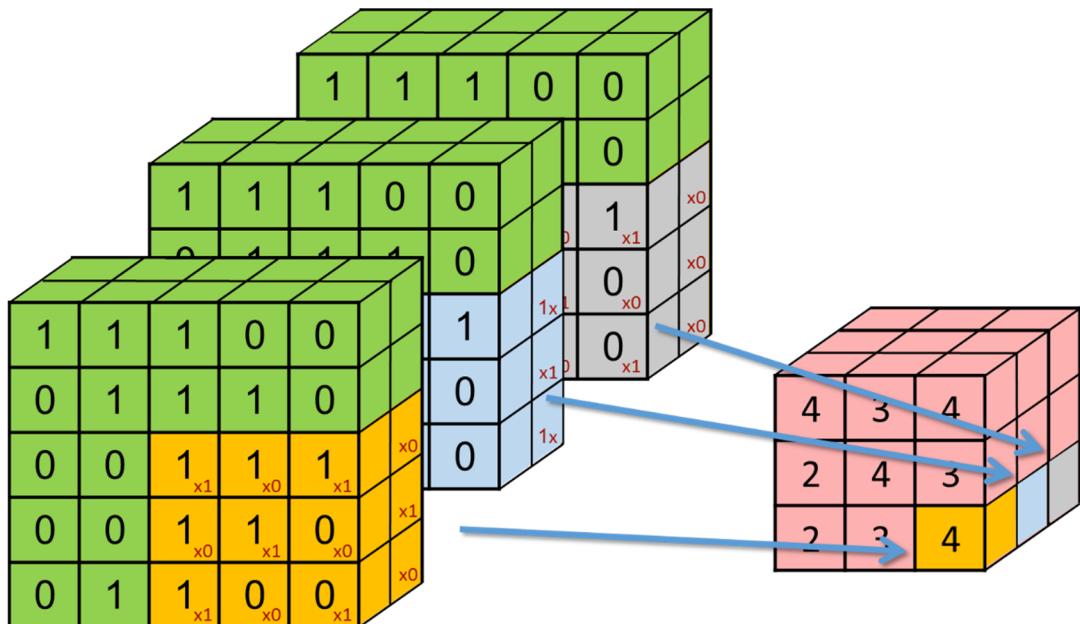
⁴ חשוב להציג שרשות נוירונים פועלות עם ערכים ממשיים ולא מספרים שלמים או בינאריים.

כפי שראינו קודם, תמונות יכולות להיות מורכבות מכמה שכבות. כיצד נראה פועלות הקונבולוציה אז?

במקרה זה כל גרעין קונבולוציה הוא בעל גובה ורוחב (3x3 כמו קודם) אך בעל עומק התלוי בעומק תמונה הקלט. בדוגמה למטה תמונה הכניסה היא בעומק 2, לכן כל גרעין גם הוא בעומק 2. עומק תמונה המוצא יהיה כמספר גרעיני הקונבולוציה. בדוגמה למטה השתמשנו ב-3 גרעינים שכן גודל תמונה המוצא הוא 3.

הערה 1 - על אף שפועלות הקונבולוציה מבוצעות על מטריצות תלת-ממדיות (המכנות טזוריים), פועלות הקונבולוציה היא דו-מימדית שכן תנועת המסננים היא רק בשני ציריהם.

הערה 2 – בדומה לשכבות לינאריות, גם כאן נוהג להוסיף גודל קבוע לכל גרעין המכונה bias.



עד כה, ראיינו 3 פרמטרים שיש לבחור כshedulerים שכבת קונבולוציה: אורך, רוחב ומספר גרעינים. נציג כעת עוד שני פרמטרים רלוונטיים.

ריפוד (padding) – בדוגמה לעיל רוחב וגובה התמונה קטן כפונקציה של גודל התמונה המקורית וגודל גרעין הקונבולוציה. את הנוסחה לחישוב הגודל בМОץ יש לחשב בתרגיל ההכנה מס' 1. ניתן לשנות את רוחב וגובה תמונה המוצא ע"י ריפוד תמונה הכניסה באפסים.

דילוג (stride) – בדוגמה לעיל, עברנו על תמונה הכניסה עם הפילטרים כאשר כל פיקסל בתמונה המוצא התקבל ע"י הזזה של גרעין הקונבולוציה ב"צעד" אחד. ניתן גם לבצע צעדים גדולים יותר ולא לבצע קונבולוציה "מלאה".

הדוגמה ויזואלית של ריפוד ודילוג ניתן לראות [בקישור הזה](#).

שכבות הורדת מידע

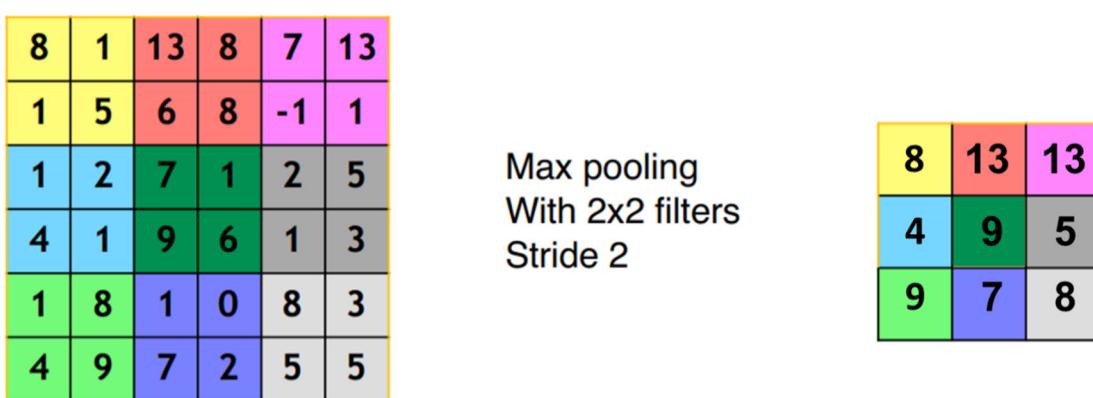
שכבה נוספת נוספת ברשותה הן שכבות הורדת מידע הנקראות pooling. שכבות אלה פועלות בצורה מקומית, בדומה לשכבות קונволוציה.

בין שכבות ה-pooling, הנפוצות ביותר הן שכבות max ו- average pooling. פעולה השכבה היא הפעלת מקסימום וממוצע על קבוצת תאים בשכבה בהתאם.

לשכבה זו 4 היפר-פרמטרים (פרמטרים שנקבעים מראש ולא נלמדים) ולא כוללת פרמטרים נלמדים כלל.

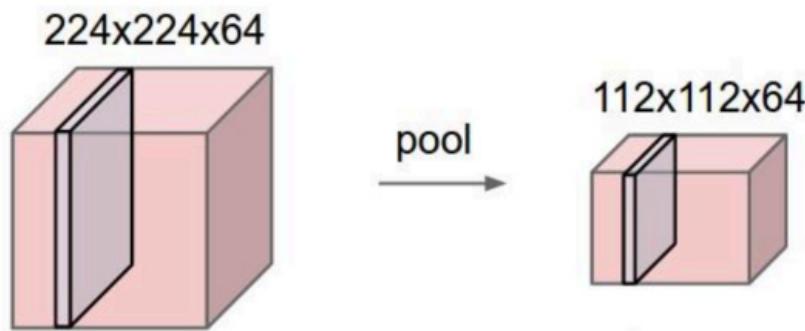
הפרמטרים הם הגודל של הפילטר (אורך ורוחב) וגודול הצעד (stride) בכל כיוון.

דוגמה מספרית לשכבה max pooling מוחשית באירור 15 – שכבת Max Pooling.



איור 15 – שכבת Max Pooling

כאשר מפעילים את השכבה על טזוזר תלת ממדי, מפעילים את הפעולה על כל שכבה בנפרד. ראו איור 16 – שכבת pooling על טזוזר תלת ממדי.



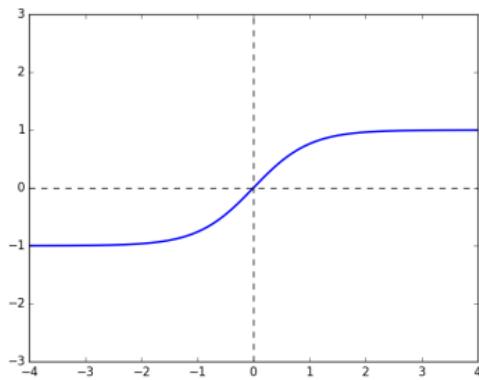
איור 16 – שכבת pooling על טזוזר תלת ממדי

סוגי אי-ליינאריות

עד כה ראיינו הכרנו שכבה ליינארית מסווג סיגמויד. בעת נציג סוגים נוספים.

סוג אחר של אי-ליינאריות שהיה 매우 נפוץ הוא :

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



שימוש לב Ci מתקיים הקשר הבא :

$$\tanh(x) = 2\sigma(2x) - 1$$

כלומר, \tanh הוא לא יותר ממיתחה והזזה של פונקציית הסיגמאoid.

שני סוגים אי הליינאריות האלה סובלים מבעיה הנקראת "בעיית הגרדיאנטים הדועכנים" (vanishing gradient).

הערך המקסימלי של פונקציית הנגזרת של סיגמוaid מתקבל עבור $0 = x$, שם

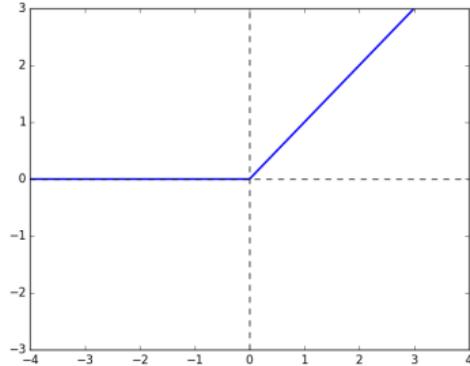
$$\frac{\partial \sigma}{\partial z}(0) = \sigma(0)\left(1 - \sigma(0)\right) = \frac{1}{2} \cdot \left(1 - \frac{1}{2}\right) = \frac{1}{4} < 1$$

כלומר, ככל שהרשות שלנו עמוקה יותר, כך נכפול את הגרדיאנט בעוד ערכים קטנים מ-1 והשגיאה תדעך יותר.

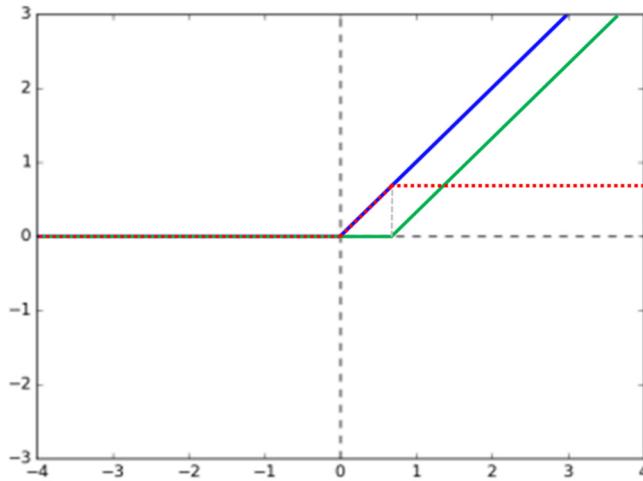
בעיה הנילישן מספר פתרונות. דרך אחת שפותרת את הבעיה היא להשתמש בשכבה אי-ליינארית אחרת.

סוג זה, הנפוץ ביותר בימינו, נקרא Rectified Linear Unit או בקיצור ReLU.

$$\text{ReLU}(x) = \max(0, x)$$



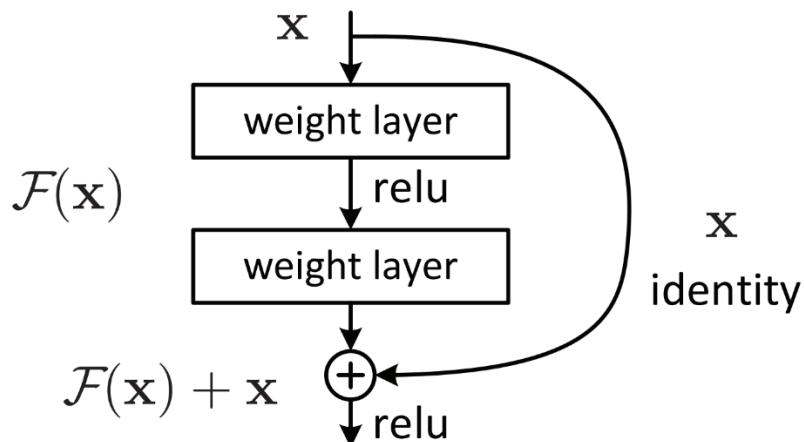
לשכבה זו צורה מאוד שונה מזו של הליינאריות הקודמת. אף על פי כן, הרכבה של שתי פונקציות כאלה יכולות בקלות לקרב פונקציות בסגנון של סיגמוaid.



בכחול ובירוק ניתן לראות שתיReLU שונות, כאשר ה-ReLU הירוק מוזז. הזויה כזו יכולה להתקבל כאשר יש איבר הטיה בשכבה הקודמת (想起你: אם מדובר בשכבה ליניארית המיצגת ע"י $y = Ax + b$ אז b הוא גורם ההטייה).
באודם מצויר הפרש בין הקו הכחול לקו הירוק. הפרש כזה יכול להיווצר מכפלת וסכום של שכבה עוקבת בה המשקولات בערכים ± 1 .
באזור בו הגראדיאנט אי-שלילי הערך שלו הוא קבוע בגודל 1. ככל מר, ככל שנתרשר יותר שכבות גודל הגראדיאנט לא ישנה.

שכבות שארית (Residual)

שכבות שארית (Residual) שגמ מכונים "חיבור מילג" (skip connection) היא שכבה נפוצה שעשויה פולה מאוד פשוטה.



שכבות שארית מעבירות את אותן הכניסה הלא ייחד בתוספת פעולה אחרת (פעולה ליניארית או פעולה קונבולוציה).

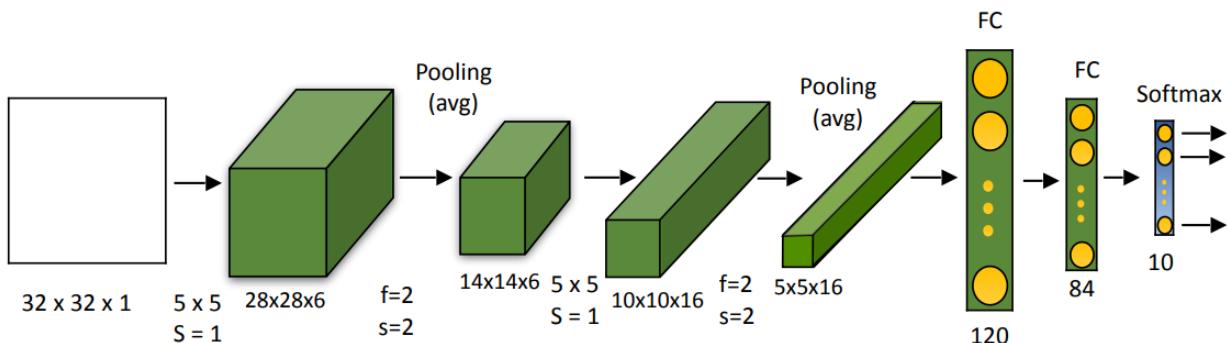
מוטיבציה אחת לשימוש בשכבות מסווג זה היא שהגראדיאנט של השכבה הוא:

$$\frac{\partial \text{residual}(x)}{\partial x} = \frac{\partial F}{\partial x} + 1$$

עויי שימוש בשכבה מסווג זה הגרדיינט לא דועך בעבר בין שכבות הרשות.

רשתות CNN סטנדרטיות

רשת קומבולוציה שהייתה בסיס לשכבות קומבולוציה עתידיות היא LeNet-5 (Y. LeCun, november 1998) :



רשת זו אומנה לזהות ולסווג ספרות כתוב יד על MNIST, אותה המשימה שראיתם בניסוי הראשוני.

הרשת מורכבת מהשכבות הבאות :

1. שכבה קומבולוציה עם 6 גרעיני קומבולוציה בגודל 5×5 (סה"כ 156 פרמטרים, כולל ה-bias של כל גרעין).
2. שכבה הורדת ממדיות average pooling עם פילטרים בגודל 2×2 וגודל צעד 2.
3. שכבה לא-ליינארית מסוג \tanh .
4. שכבה קומבולוציה עם 16 גרעיני קומבולוציה בגודל 5×5 (סה"כ 2416 פרמטרים).
5. שכבה הורדת ממדיות average pooling עם פילטרים בגודל 2×2 וגודל צעד 2.
6. שכבה לא-ליינארית מסוג \tanh .
7. שכבה קומבולוציה עם 120 גרעיני קומבולוציה בגודל 5×5 (סה"כ 48120 פרמטרים).
8. שימושו לב שהגודל המרחביב של תמונות הכניסה לשכבה זו הוא 5×5 לעומת גודל הפילטר הוא גודל התמונה! על כן במקרה זה פועלות הקומבולוציה למעשה שקופה לחולותין לפעולה של שכבה ליינארית (כפל במטריצה).
9. שכבה ליינארית בגודל 84 (מספר פרמטרים 10164).
10. שכבה ליינארית בגודל 10 (מספר פרמטרים 850).
11. שכבה שגיאה מסווג Negative Log Likelihood.

שיטות אופטימיזציה מתקדמות

זכיר את אלגוריתם האימון בו השתמשנו עד כה : gradient descent . בහינתו פונקציה $f(x)$, בכל צעדים, נחשב את הגרדיינט של הפונקציה ולאחר מכן נוע בכוון הגרדיינט כפול גודל הצעד.

$$g_k = \nabla f(x_k)$$

$$x_{k+1} = x_k + \eta_k g_k$$

כעת נציג מספר וריאציות על שיטה פשוטה זו שנפוצות באימון רשותות.

1. מומנטום

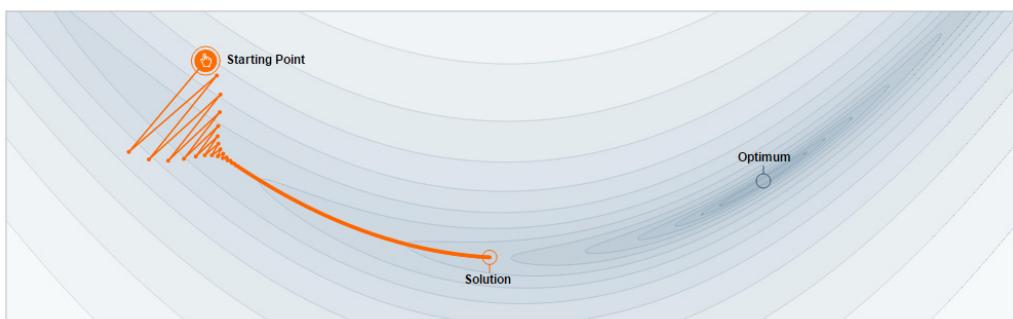
שיטות מומנטום, או בשמה המלא stochastic gradient descent with momentum , משנה את צעד העדכון כדי להתחשב בקצב שינוי הפרמטרים במרחב הפרמטרים.

$$v_{k+1} = \alpha v_k - \eta \nabla f(x_k)$$

$$x_{k+1} = x_k + v_{k+1}$$

האיבר α הוא קבוע קרוב ל-1 (בדרך כלל 0.9) המשמש לקבוע את מידת ההתחשבות בגרדיינטים של העבר.

משמעות זה שקול לדחיפת כדור במורד מדרון (הגרדיינט), כאשר הוא עם הזמן צובר תאוצה (מומנטום) אבל פועל עלייו כח חיכוך עם האויר (גודל הצעד). ויזואלית ההבדלים נראה כך :



אינטרואיציה פיזיקלית

לאלו מכמ שמרגשים בנה עם משוואות תנועה ניוטוניות, מצורף ההסבר מטה.

נתבונן בגרסה רציפה של gradient descent :

$$\frac{dx}{dt} = -\eta \nabla_x E(x)$$

כאשר $E(x)$ הוא פוטנציאל חיצוני (שנגזרתו היא הכוח הפועל על מסה) ו- $\frac{dx}{dt}$ מבטא את המהירות (קצב שינוי הפרמטרים).

הגרסה הרציפה של מומנטום מתאימה לשווה הניוטונית המתארת תנועת מסה ו- בtower

: $E(x)$ צמיגי עם מקדם חיכוך μ תחת השפעת כוח בעל פוטנציאל

$$m \frac{d^2x}{dt^2} + \mu \frac{dx}{dt} = -\nabla_x E(X)$$

ע"י דיסקרטיזציה של המשווה נקבל :

$$m \frac{x_{t+\Delta t} + x_{t-\Delta t} - 2x_t}{(\Delta t)^2} + \mu \frac{x_{t+\Delta t} - x_t}{\Delta t} = -\nabla_x E(x)$$

$$x_{t+\Delta t} - x_t = -\frac{(\Delta t)^2}{m + \mu \Delta t} \nabla_x E(x) + \frac{m}{m + \mu \Delta t} (x_t - x_{t-\Delta t})$$

הביטוי הנ"ל זהה לביטוי המקורי של המומנטום.

2. Adaptive Gradient (AdaGrad)

AdaGrad היא שיטתgradient descent בה לכל פרמטר קצב לימוד מסוילו הנקבע על סמך היסטוריית הגדיינטים שלו. אלגוריתם זה נותן קצב לימוד גבוה לפרמטרים שימושיים לעיתים רחוקות וקצב לימוד נמוך לפרמטרים שימושיים לעיתים קרובות. משווהות האלגוריתם נראה כך :

$$r_{k+1} = r_k + g \odot g$$

$$x_{k+1} = x_k - \frac{\eta}{\epsilon + \sqrt{r}} \odot g$$

כאשר \odot מייצג מכפלה איבר-איבר, ϵ הינו גודל קטן שנועד למנוע בעיות נומריות.

בעיה אפשרית של שימוש ב-Adagrad היא העובדה שהווקטור r הולך וגדל לאורץ האימון ולא מאפשר "לשכוח" גדיינטים שראתה בעבר הרחוק.

3. RMSprop

שיטת RMSprop מתמודדת עם הבעיה הנ"ל ע"י חלפת העדכון של r בממוצע נع של r כך :

$$r_{k+1} = \rho r_k + (1 - \rho)g \odot g$$

$$x_{k+1} = x_k - \frac{\eta}{\epsilon + \sqrt{r}} \odot g$$

כאשר $0.9 = \rho$ בד"כ.

4. Adaptive Moment Estimation (Adam)

Adam הוא אלגוריתם שמנסה שלב בין מומנטום לבין שיטות גרדיאנט אדפטטיביות :

$$\begin{aligned} m_{t+1} &= \beta_1 m_t + (1 - \beta_1) g \\ v_{t+1} &= \beta_2 v_t + (1 - \beta_2) g \odot g \\ x_{k+1} &= x_k - \frac{\eta}{\epsilon + \sqrt{v_{t+1}}} \odot m_{t+1} \end{aligned}$$

m_t ו- v_t הם שערוכים של המומנטום מסדר ראשון ושני (توزלת ושונות) של הגרדיינטם בהתאם, ומכאן נובע השם.

אבל לשיטה הניל יש בעיה : מכיוון שמאתחלים את הוקטורים m_t ו- v_t לוקטוריים אפסים, בהמשך תהליך האימון ל- m_t ול- v_t יש נטייה להישאר קרובים ל-0. כדי להתמודד עם הבעיה זו מבצעים את התיקון הבא :

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1 - \beta_1^t} \\ \hat{v}_t &= \frac{v_t}{1 - \beta_2^t} \\ x_{k+1} &= x_k - \frac{\eta}{\epsilon + \sqrt{\hat{v}_{t+1}}} \odot \hat{m}_{t+1} \end{aligned}$$

מעשית ברוב הרשותות המודרניות משתמשים ב- Adam או ב- SGD + Momentum.

רגוליזציה

ברשותות מודרניות יש מיליון פרמטרים, לעיתים יותר ממספר הדוגמאות שרואים באימון! כתוצאה לכך, רשותות נוטות לעשות overfitting (ראו <https://en.wikipedia.org/wiki/Overfitting>). כלומר להתאים את עצמן בצורה כמעט מושלמת לדוגמאות שהן ראו על חשבו ההכללה לדוגמאות שלא רואו בסטו האימון.

רגוליזציה מוגדרת להיות כל שינוי שעושים ברשות שנוועדה להקטין את שגיאת ההכללה אך לא את שגיאת האימון.

שיטות רגוליזציה בד"כ מעודדות מודלים פשוטים יותר בהתאם לעיקרונו Occam's razor. דוגמה לרגוליזציה שראינו היא שימוש ברשותות קוונבולוציה במקום רשותות לינאריות fully connected שמשתמשות במידע שיש לנו על מבנה של תכונות טבעיות ועל האופן שבו ניתן לחץ מאפיינים שלחן.

כעת נציג מספר כלים מארכז הכלים שיש לנו כדי להתמודד עם התופעה.

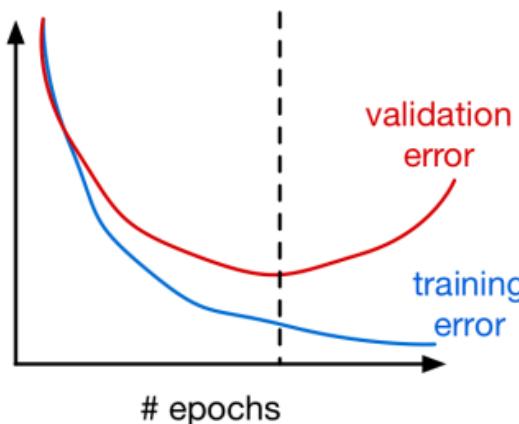
1. Early Stopping

אחד הדרכים פשוטות להימנע מ- overfitting היא לעצור את תהליך האימון ברגע שאחנו לא רואים יותר שיפור ביכולת ההכללה של הרשות.

ניתן לזהות נקודה זו ע"י בוחנת השגיאה של הרשות על סט הולידציה וערכת האימון כאשר שגיאה זו גדולה.

ניתן גם לבצע "עכירה בדיעבד". כלומר, להמשיך לאמן, ואז להשתמש במודל כפי שהוא מכבו בנקודת שוויהינו בטובה לעכירה.

שימו לב - כלי רשות הנוירונים של MATLAB מבצע Early stopping באופן דיפולטי. השיטה בה הם פועלים היא בדיקה של המגמה של סט הולידציה - במידה וה-error validation גדול כמעט ממספר מסויים של בדיקות, התהליך נעצר.



איור 17 – הנקודה באימון בה נחליט לעצור כדי להימנע מ- overfitting

2. פחות פרמטרים

השיטה הטריויאלית והמתבקשת היא הקטנת מספר הפרמטרים של הרשות ושימוש ברשות קטנה יותר. למשל ע"י מעבר לשכבות קונבולוציה, הקטנת מספר המסלנים, פחות שכבות ברשות.

3. Weight Decay

רגוליזציה מסווג weight decay, เชגמ מכונה ridge או L2, מגבילה את מרחב האפשרויות של משקלות הרשות ע"י "הענשה" על שימוש במסקלות גדולים מדי.

עבור פונקציית השגיאה הריבועית (כמו ב-*Adaline*), פונקציית המחיר החדש תיראה בצורה הבאה :

$$Err(f) = \frac{1}{n} \sum_{i=1}^n (w^T x_i - y_i)^2 + \frac{\alpha}{2} \|w\|_2^2$$

עדכון המשקלות, עפ"י הגרדיאנט החדש ייראה כך :

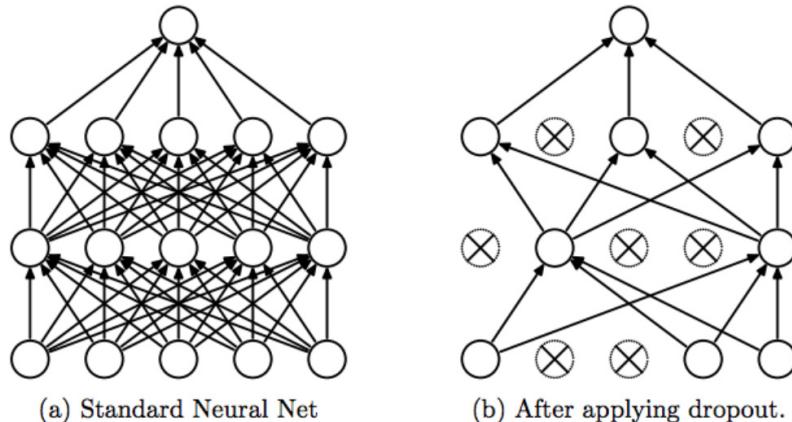
$$w^{t+1} = w^t - \eta_t \left(x \left(w^{t^T} x - y \right) - \alpha w \right) = (1 - \eta_t \alpha) w - \eta_t \left(w^{t^T} x - y \right)$$

קיבלו כי עדכון המשקלות זהה לעדכון הקודם למעט דעיכה של המשקלות בפקטור $(1 - \eta_t \alpha)$. weight decay .

4. Dropout

Dropout היא שיטת רגוליזציה נוספת לרשותה בה "מכבים" נוירונים בכל איטרציה אקראית.

עושים זאת ע"י הגדרת מטריצת מסיכה בינהריה בה כל איבר מתפלג ברנולי (התלת מטבע) עם פרמטר שנקבע ע"י המשתמש. ככלומר בכל איטרציה נירון אחר "נכח". האינטואיציה לשימוש בשיטה זו היא שכעת על הרשות ללמידה לסוג באפשרויות מסווגים שונים בראש והיא לא יכולה להסתמך על מספר קטן של נוירונים. נציג שאות שיטת **dropout** מפעילים במהלך האימון בלבד.



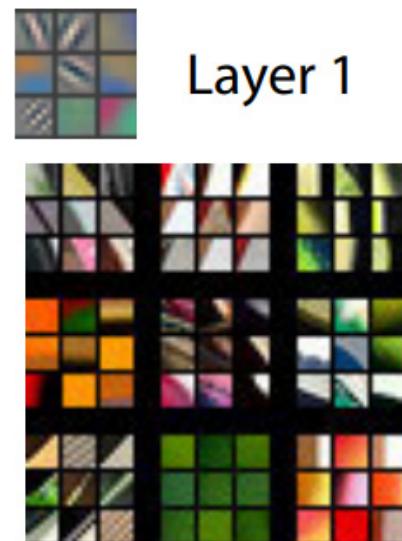
איור 18 – רשת לינארית עם ובלי **dropout**

5. Data Augmentation

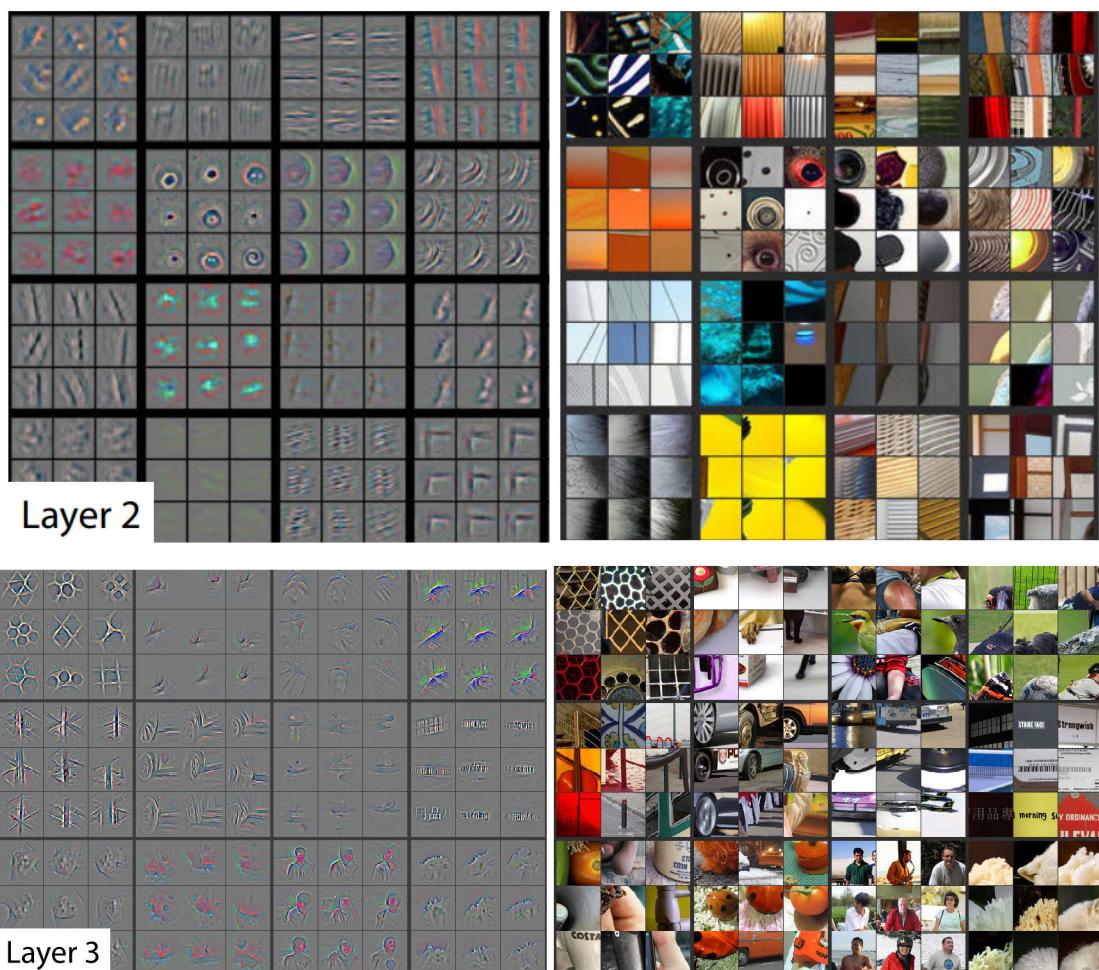
הדרך הטובה ביותר לשפר את יכולת הכללה של המודל היא לאסוף עוד דוגמאות! אך רוב זה לא אפשרי ואנחנו מוגבלים בכמות הדוגמאות שיש לרשותנו (או שהתחליק מאוד יקר). **Data augmentation** מציע להגדיל את אוסף הדוגמאות שלרשוטנו ע"י ביצוע שינויים בדוגמאות הקיימות כך שיהו דוגמאות אחרות. דוגמאות לשינויים נפוצים לתמונות כוללות: טרנסלציה (הזזה), היפוך, סיבוב והוספת רעש.

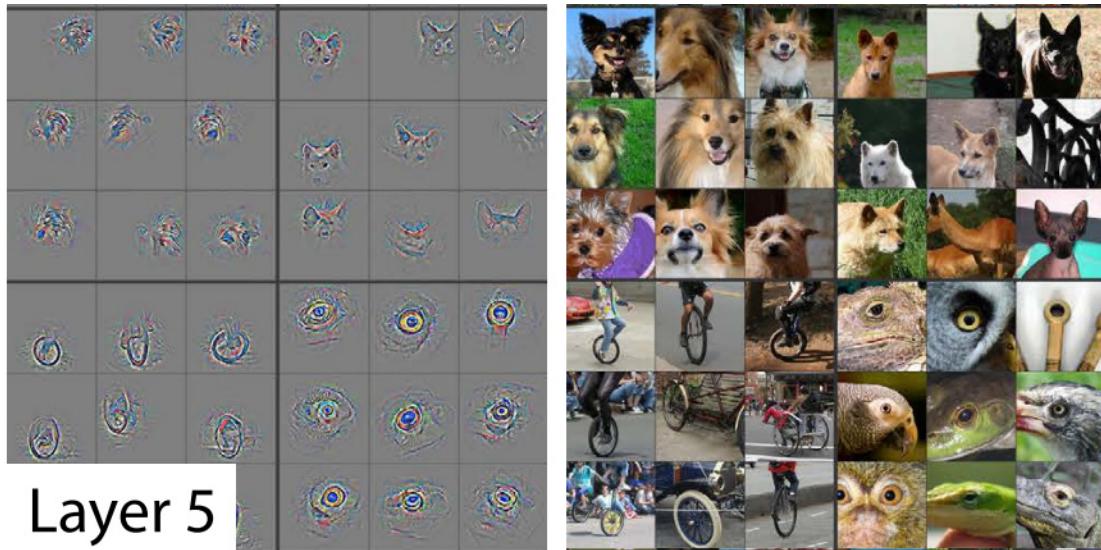
מה נלמד בשכבות הפנימיות בראש

מטרז ניסינו להבין כיצד רשתות נוירוניים عمוקות מצלחות לפתרן בעיות מורכבות (בעיקר בעיות תמונה), החלו לחזור מה נלמד בכל שכבה בראש. בתהיליך זה גילו כי שכבות נמוכות לומדות לזהות מאפיינים בסיסיים של תמונות כמו שפות ופינות, שכבות אמצעיות מזוהות טקסטורות וצורות גאומטריות ושכבות גבהות יותר מזוהות עצמים מורכבים יותר כמו עיניים, פרצופים של בני אדם ושל בע"ח. ויזואлизציה של שכבות בעומקים שונים בראש שאומנה על סיוג תמונות ניתן לראות באיור 19 – אקטיבציות של שכבה ראשונה בראשת קלסיפיקציה. מימין רואים את תמונות המקור שהוזנו לראש ומשמאלו את האקטיבציות (פלט של שכבה בראשת לאחר שכבת אי הלינאריות) של שכבות בעומקים שונים בראש.



איור 19 – אקטיביזיות של שכבה ראשונה בראשת קלסיפיקציה





המעוניינים למדוד עוד על הנושא (או סתם לצפות בתמונות מרהיבות בצורה אינטראקטיבית) מוזמנים להיכנס [לכאן](#).

Transfer learning

מודדר כז:

Transfer learning and domain adaptation refer to the situation where what has been learned in one setting ... is exploited to improve generalization in another setting

— Page 526, Deep Learning, 2016.

כפי שראינו בסעיף הקודם, בשכבות שונות של הרשת נלמדים מאפיינים שונים של הקלט שלנו. אם ניקח את הרשת ונקטע אותה בשכבה ביניים, למעשה קיבלנו פונקציה שמטילה תמונה למרחב אחר בה יש **משמעות סמנטית** שנייה לנצל לטובת סיוג.

אך ניתן להשתמש באוותה הפונקצייתית כדי להטיל את מרחב הדוגמאות למרחב אחר, בו נוכל לפתור בעיות סיוג ורגסיה שונות מאשר המשימה המקורית אליה אימנו את הפונקציה.

כיצד עושים זאת? את השכבות המאוחרות של הרשת שלמדו לסוג דוגמאות למשימה ספציפית, נחליף בשכבות חדשות – אותן נאמן על דוגמאות חדשות התואמות את המשימה החדשה.

בחלק השני של הנסיוני נדגים כיצד ניתן לקחת רשת שהתאימה לביעית קלאסיפיקציה של תמונות טבעיות ולפתור באמצעותה בעיה של קלאסיפיקציה של תמונות של קינוחים.

מפגש שני

שאלות הינה

1. נתונה תמונה כניסה לרשת עם מימדים h_{in}, w_{in}, c_{in} .

שכבה הקובולוציה הראשונה בעלת הנתונים הבאים :

- מסננים בגודל k_h, k_w
- מספר המנסנים הוא d
- אין ריפוד או דילוג.

כתבו ביטויים ל- $h_{out}, w_{out}, c_{out}$, מימי תמונה המוצא משכבה הקובולוציה כפונקציה של הפרמטרים לעיל.

2. שיטות אופטימיזציה מתקדמות

בහינתן הפונקציה $f(x) = x^2$ ונקודת התחלת $x_0 = -1$, הדגש שלוש איטרציות של gradient descent עם מומנטום.

א. עבור גודל צעד $\eta = 0.01$.

ב. עבור גודל צעד $\eta = 2$.

Weight Decay .3

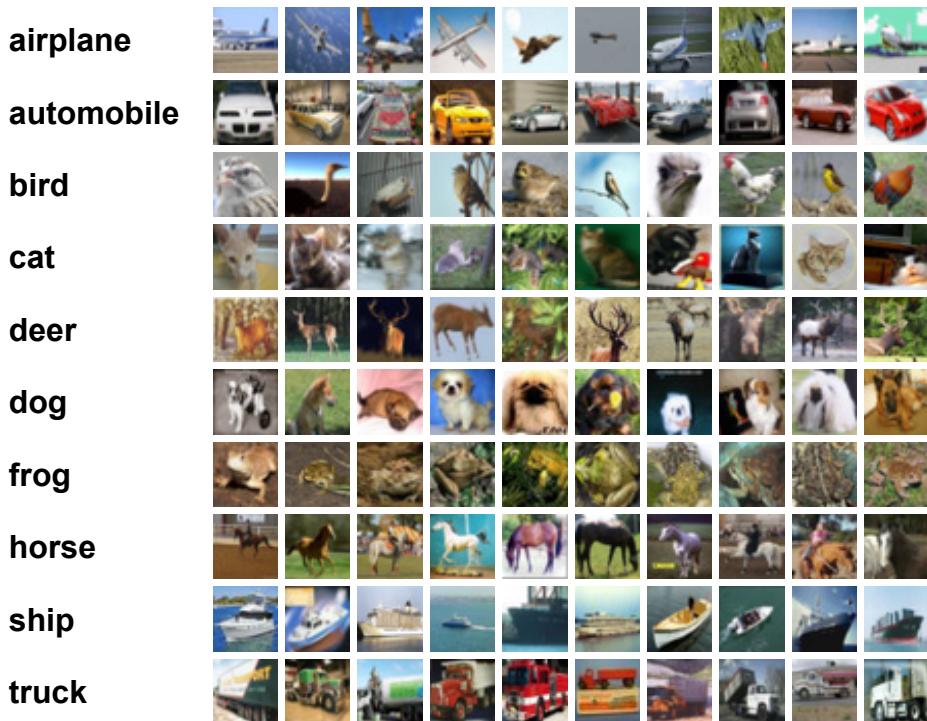
באיזה אופן משפיעה רגולריזציית L2 על המשקלות? התנייחסו לפרמטר α בערכי קיצון $(\alpha \rightarrow 0, \alpha \rightarrow \infty)$.

4. מהם היתרונות של CNN על פני fully connected במשימות סיווג תמונה?

מפגש שני – מהלך הניסוי

חלק א' – סיווג תמונות CIFAR10

בחלק זה נתמודד עם משימת סיווג תמונות ממאגר CIFAR10. מאגר זה מכיל תמונות בקטגוריות שונות, כאשר כל אחת מתויגת על ידי הקטגוריה המתאימה. הקטגוריות הן:



אייר 19 - הקטגוריות במאגר cifar10 ודוגמאות לתמונות מכל קטgorיה
לköח מותך <https://www.cs.toronto.edu/~kriz/cifar.html>

נרצה לבנות רשת ניורונים המסוגת תמונות לקטגוריות המתאימות ב-CIFAR10. לשם כך נבנה ארכיטקטורה המשלבת שכבות קונבולוציה ושכבות לינאריות.

- ❖ הריצו את החלק הראשון בקוד cifar10ClassificationDL אשר מוריד את CIFAR10. עדכנו את הנתיב לתיקייה המתאימה.

בשלב התאמת הפרמטרים של הרשת, מטעמי חישכון בזמן, נסוג רק 4 מחלקות מתוך 10 המחלקות של CIFAR10. בסוף התאמת הפרמטרים נבצע אימון סופי על 10 המחלקות.

- ❖ עברו על הקוד והשלימו את השכבות הרלוונטיות לפי ההוראות. השכבות הנחוצות לחלק זה נמצאות בספר א' – Matlab Layers חלק א', נספח ב' – Matlab Layers חלק ב'.

בחלק הקודם ראיינו את השפעת גודל הצעד על התוכניות הרשת. שלב בחירת הצעד הוא שלב חשוב באימון כל רשת, ומכיון שעסקנו בו רבות בחלק הקודם בחלק הזה נניח שידעו גודל צעד שעובד טוב ונשותמש בו.

הערה: ישן אופטימיזציות שונות הגורמות לירידה בגודל הצעד ככל שמתקדמים בתהליכי הלמידה, ובכך מביאות לדיווק גבוה יותר בהתכנסות. חלק זה של המעבדה השתמש ב贊א אופטימיזציה, אשר מקטינה את גודל הצעד בפקטור קבוע כל כמות epoch-ים שנקבעת מראש. ניתן לראות את השימוש בחלק הבא באופציות האימון: 'LearnRateSchedule', 'LearnRateDropFactor', 'LearnRateDropPeriod'.

נבחן את השפעות הוסף רגולרייזציה על הרשת.

כפי שאתם יכולים לראות בקוד, כבר קיימות רגולרייזציות 2.

❖ הריצו את הרשת עם הפרמטרים הנוכחיים וצרפו את תוצאות הריצה.

❖ עבור סעיף זה בלבד, הגדילו את ערך רגולרייזציה 2 ל-0.5. צרפו את תוצאות הריצה.

1. מה קורה לתהליכי התכנסות? מה הסיבה?

❖ הוסיפו לרשת שכבת dropout ייחידה, בסוף הקונולוציות מיד לפני שכבות ה-.
fullyConnected

2. הסבירו מהי משמעות ערך ה-dropout probability ב-
.dropout

❖ הריצו את הרשת עם לפחות 3 ערכים שונים של dropout probability בטווח 0-0.5, כל פעם במשך 25 אפוקים.

3. מה ההשפעה של dropout על התכנסות? הסבירו והוסיפו נתונים וגרפים מתאימים. בחרו את הערך המתאים ביותר והמשיכו אותו להלאה.

cut, נסיף Data augmentation.

4. איזה סוג Data augmentation מתאים כאן? הסבירו בקצרה בעזרת דוגמאות.

❖ הסתכלו בתחילת הקוד על השורות אשר בהערות המתאיםות לסעיף ה-Data augmentation. הוסיפו אותן לקוד, והגדירו את ה-Data augmentation אשר נראה בעיניכם מתאים. היעזרו בתיעוד של MATLAB עבור האופציות הקשורות ב-imageDataAugmenter. שימו לב שעיליכם להגדיר את imageDatastore כפונקטר `augmentedImagestore`. הריצו אימון חדש וצרפו את התוצאות לדוח.

5. איך ה-Data augmentation השפיע על התוצאות?

❖ בטלו את הגבלה ל-4 מחלקות מתוך ה-10 והריצו אימון חדש עבור סיוג לכל המחלקות. צרפו לדוח שלכם את התוצאות המתקבלות.

חלק ב' – Transfer Learning

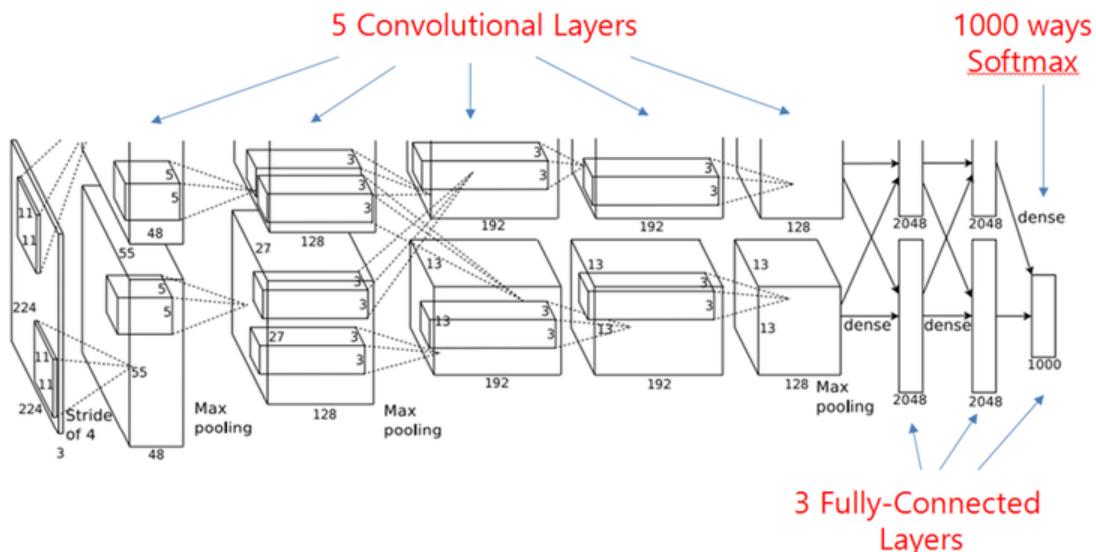
בחלק זה נאמן רשת נוירוניים לשיווג סוגים קיינוחים. Dataset המשמש למשימה זו מורכב מתמונות, ותיגו של כל תמונה אשר מעיד על המאכל בתמונה. Dataset המשמש ב-Transfer Learning. לשם כך, נשתמש ב-Transfer Learning – ניקח רשת שאומנה מראש לשיפור את ביצועי המערכת. מוגדרת רשת חדשה אשר ניתנת לכטט בתרגיל זה הינה קטנה, ונרצה לנצל כל ידע מוקדם שקיים לנו על מנת לשפר את ביצועי המערכת. Dataset המשמש לשיפור תמונה (בדומה לרשת מהסעיף הקודם), וنبנה על בסיס הרשת המאומנת מראש לשימוש לשיווג תמונות (ב證據 לרשת מהסעיף הקודם), ונבנה על בסיס הרשת המאומנת מודל חדש אשר יורח בפתרון הבעיה הייעודית החדשה. לבסוף, נאמנו את המודל החדש בעזרת Dataset החדש ונבחנו את תוכנות הרשת החדשה שבנוינו ואימנו.

Dataset ניתן למצוא באינטרנט רשתות שונות פטור בעיות שונות, אשר ניתנות להורדה. למשל, עבור עיתת הקלסיפיקציה עמה התמודדנו בסעיף הקודם ניתן למצוא רשתות בארכיטקטורות שונות (AlexNet, GoogLeNet ועוד) המאומנות מראש להורדה זמינה.

בדוגמה זו השתמש ברשת AlexNet לשיווג תמונות, אשר אומנה על מאגר חלקי מתוך ImageNet (מידע נוסף על מאגר ImageNet ניתן למצוא [כאן](#)).

❖ גירסה מאומנת של AlexNet עבור MATLAB ניתנת להורדה [כאן](#).

Alexnet היא הרשת אשר הובילה למחפכה ב-2012 כאשר הצלחה לשׂווג את מאגר ImageNet בדיק של מעל 10% מכל שאר המתחרים במשימה. הרשת מורכבת מ-5 שכבות קונבולוציה ו-3 שכבות של מילוי. על מבנה הרשת וההידושים שבו תוכלו לקרוא [פה](#). Fully-Connected



איור 20 - מבנה הרשת Alexnet

על מנת להתאים את הרשת לשימוש השיווג שלנו, ניקח את הרשת המאומנת מראש ו"ינקפייה" את השכבות שלא פרט ל-3 שכבות האחרונות, אותן נחליף לשכבות חדשות ונאמנו על המאגר החדש עבור המשימה החדשה. לבסוף, הרשת שנקבל תהיה מורכבת מ-22 שכבות עם משקלות קבועות (בצבע כחול), ו-3 שכבות אשר המשקלות שלהן יילמדו בתהליכי אימון מחדש (בצבע אדום).

```

' 1 data'    Image Input
' 2 conv1'   Convolution
' 3 relu1'   ReLU
' 4 norm1'   Cross Channel Normalization
' 5 pool1'   Max Pooling
' 6 conv2'   Convolution
' 7 relu2'   ReLU
' 8 norm2'   Cross Channel Normalization
' 9 pool2'   Max Pooling
' 10 conv3'  Convolution
' 11 relu3'  ReLU
' 12 conv4'  Convolution
' 13 relu4'  ReLU
' 14 conv5'  Convolution
' 15 relu5'  ReLU
' 16 pool5'  Max Pooling
' 17 fc6'    Fully Connected
' 18 relu6'  ReLU
' 19 drop6'  Dropout
' 20 fc7'    Fully Connected
' 21 relu7'  ReLU
' 22 drop7'  Dropout
' 23 fc8'    Fully Connected
' 24 prob'   Softmax
' 25 'output' Classification Output

```

בתיקייה הרלוונטייה לחלק זה מיצורפים לכמ שני קטעי קוד חלקיים -

- `TransferLearning.m` – הפעלת המודול לאימון ובדיקת הרשות למשימה החדש
- `freezeWeights.m` – פונקציה מקבלת אוסף שכבות ומחזירה גירסה "МОוקפת" של השכבות.
- א. מהם היתרונות של Transfer Learning? באילו סוגים של שימוש ניתן להשתמש בשיטה זו?
- ב. אילו מאפיינים תצפו שייוו לפט של ריצה עם שימוש ב-`Transfer Learning` לעומת ריצה המאמנת רשות שלמה מנוקודת מוצאת נקייה?
- ❖ השלימו את שורות הקוד במקומות המתאימים בקובץ `.TransferLearning.m`.
- ❖ הוסיפו לדוח שלכם את פלט ההערכת והגרף המתתקבל.
- ❖ השתמשו במידע שרכשתם במעבדה זו על מנת לנסות למקסם את תוצאות ה-`Transfer Learning`. שחקו עם הארכיטקטורות והפרמטרים באיזו צורה שתחפזו. הסבירו בדוח מה ניסיתם לעשות וצרפו תוצאה שקיבלתם.

נספח א' חלק א' Matlab Layers

טבלה מסכמת של השכבות הנחוצות לשימוש במקיש א' (לקוח מהטייעז באתר של MathWorks).

Input Layers

Function	Description
imageInputLayer	An image input layer inputs images to a network and applies data normalization.

Learnable Layers

Function	Description
fullyConnectedLayer	A fully connected layer multiplies the input by a weight matrix and then adds a bias vector.

Activation Layers

Function	Description
reluLayer	A ReLU layer performs a threshold operation to each element of the input, where any value less than zero is set to zero.

Normalization and Dropout Layers

Function	Description
dropoutLayer	A dropout layer randomly sets input elements to zero with a given probability.

Output Layers

Function	Description
softmaxLayer	A softmax layer applies a softmax function to the input.
classificationLayer	A classification output layer holds the name of the loss function the software uses for training the network for multiclass classification, the size of the output, and the class labels.
regressionLayer	A regression output layer holds the name of the loss function the software uses for training the network for regression, and the response names.

נספח ב' – חלק ב'

טבלה מסכמת של השכבות הנחוצות לשימוש במפגש ב' **בנוסף** לשכבות המוצגות בספח א' (לקוון מהתייעז באתר של MathWorks).

Function	Description
<code>convolution2dLayer</code>	A 2-D convolutional layer applies sliding filters to the input. The layer convolves the input by moving the filters along the input vertically and horizontally and computing the dot product of the weights and the input, and then adding a bias term.

Pooling Layers

Function	Description
<code>averagePooling2dLayer</code>	An average pooling layer performs down-sampling by dividing the input into rectangular pooling regions and computing the average values of each region.
<code>maxPooling2dLayer</code>	A max pooling layer performs down-sampling by dividing the input into rectangular pooling regions, and computing the maximum of each region.
<code>maxUnpooling2dLayer</code>	A max unpooling layer unpools the output of a max pooling layer.