

1. Introduction

1.1 What is Machine Learning?

1.1.1 The Basic Concept

Artificial Intelligence (AI)

בינה מלאכותית הינו תחום בו תוכנות מחשב או מנגנון טכנולוגי אחר מחקה מגנון חישיבה אנושי. בתחום רוחב זה יש רמות שונות של בינה מלאכותית – יש מערכות שמסוגלות ללמידה דפואית התנהגות ולהתאים את עצמן לשינויים, ואילו יש מערכות שאמן מחקות מגנון חישיבה אנושי אך הן לא מתוכננות מעבר להה שתוכנתן אותן אוטומטית. שואב רוחבי היודע לחשב את גודל החדר ואת מסלול הנקיי האופטימלי פועל לפי פרוצדורה ידועה מראש, ואין בו תחכים מעבר לתוכנות הראשוני שלו. לעומת זאת תוכנה היודעת לס肯 רעשם באופן מסתגל, או להמליץ על שירים בנגן מזיקה בהתאם לسانון של המשמש, משתמשת בבינה מלאכותית ברמה גבוהה יותר, כיוון שהן חומרות עם הזמן דברים חדשים.

המונה בינה מלאכותית מתייחס בדרך כלל למערכת שמקהה התנהגות אנושית, אך היא שגרתית, לא לומדת משנה חדשה, ועושה את אותן הדברים כל הזמן. מערכת זו יכולה להיות משוכללת ולהשיב דברים מסוימים ואף להפסיק מסקנות על דוגמאות חדשות שהיא מעולמת לא ראתה, אך תמיד עברו אותה הקולט (Intent), שהיא אותו הפלט (Output).

נראה לדוגמא מערכת סטרימינג של סרטים, למשל Netflix, יינתן לבנות מגנון המלצות הבניי על היסטוריית השימוש של הלקוחות שלו במערכת – איזה סרטים הם אוהבים, איזה ז'אנרים וموוי. כדי שמעט צופים ומעט סרטים, ניתן לעשות זאת באופן יידי – למלא טבלאות של הנתונים, לנתח אותן ידנית ולבנות מערכת חוקים שמהווה מענו המלצות מבסס AI. ניקח לדוגמא אדם שצופה ב"פארק היורה" וב"איידנה ג'וסט" – סביר שהמערכת תמלץ לו לצפות גם ב"פולטרג'יסטים". אדם שצופה לעומת זאת ב- "אהבה בין הכרמים" ו"הבית על האגם", ככל הראה כדאי להמליץ לו על "הגשרים של מוז מדיסון".

מערכת זו יכולה לעבוד טוב, אך במקרה מסוימת כבר לא ניתן לנתח אותה כפרוצדורה מסודרת וכואסfic של חוקים ידוע מראש. מאגר הסרטים גדול, גושים סוגיים גושים של סרטים (כמו למשל סדרות, תוכניות ריאליטי ועוד) ובמושג רוצים להתייחס לפתרונות נוספים – האם הצופה ראה את כל הסרט או הפסק באמצע, מה גיל הצופה ועוד. מערכת הבניה באופן קלאסי אינה מסוגלת להתמודד עם כמויות המידע הקיימות, ומאות הכללים שנדרש לחישוב עליהם מראש היא עצומה ומורכבת לחישוב.

נתבונן על דוגמא נוספת – מערכת לניתוח רכב. ניתן להגדיר כלל פשוט בו אם משתמש יעצה מטל אביב ורוצה להגיע לפתח תקווה, אזי האלגוריתם יתקח אותו דרך מסלול ספציפי שבחר מראש. מסלול זה לא מתחשב בפתרונות קרטיסים כמו מה השעה, האם יש פקקים או חסימות ועוד. כמות הפתרונות שיש להתייחס אליהן איננה ניתנת לטיפול על ידי מערכת כללים ידועה מראש, וגם הפונקציונליות המתאפשרת היא מוגבלת מאוד – למשל לא ניתן לחזות מה תהיה שעת ההגעה וכדומה.

Machine Learning (ML)

למייד מוכנה הוא תחת תחום של בינה מלאכותית, הבא להתמודד עם שני האתגרים שתוארו קודם – היכולת לתוכנתן עם מערכת על בסיס מסוות של נתונים ופרמטרים, וחיזוי דברים חדשים כתלות בפתרונות רבים שיוכלים להשתנות עם הזמן. מגנוני ML מנתחים אדריות של דאטא ומנסוחה להציג לאיזו תוצאה. אם מדובר באפקטיביות ניוט, המערכת תתבצע את כל ווותם הפוטריים ונססה לחשב את מושך הסעודה המשוער. נניח והיא חזרה 20 דקות כסעה. אם בסופו של דבר הנסעה ארוכה 30 דקות, האלגוריתם יסתה להבין איזה פקטור השתנה במהלך הנסיעה הוא נכשל בחיזוי (למשל: הcabישן בן ארבעה ונתיבים, אבל במקטע מסוים הוא מוצטמצם לאחד וזה מייצר עיכוב, וזה עיכוב קבוע בהרבה שעות רגילה ולא פסק ארקא). ברגעון סופי מקרים כאלה, האלגוריתם "מבנה" שהוא טועה פשוט יתקן את עצמו ויכניס למערך החישובים גם פקטור של מספר נתיבים ויוריד אולי את המשקל של הטופופטרורה בחוץ. וכך באופן חרדיי האלגוריתם שוב ושוב מקבל קלט, מוציא פלט, ובודק את התוצאות הסופיות. לאחר מכן הוא בזוק היכן הוא טעה, משנה את עצמו, מתקן את המשקל שהוא נותן לפקטורים שונים ומסתכל מושפעה לנשיאה.

במערכות אלה הקולט נשאר לכואורה קבוע, אבל הפלט משתנה – עבור זמני יציאה שונים, האלגוריתם יעריך זמני נסעה שונים, כתלות במוגן הפתרונים הרלוונטיים.

מערכות ML משמשות את כל רשותות הפרסום הגדולות. כל אחת מנסה בדרך כלל להחזות למשך זמן, איזה משתמש שהקлик על המודעה צפי שיבצע רכישה. הפלטפורמות השונות מנוטות להזות כונה (Intent) על ידי למידה

מניסין. בהתחלה הן פשוטו ויחשו על פי כמה פקטוריים שהזנו להם על ידי בני אדם. נניח, גугл החלטה שמי שצופה בסרטוני יטivist של **Unboxing** הוא בא-Intent גבוה של רכישה. בהמשך הדרך, בהנחה והמשתמש מבצע רכישה כלשה, האלגוריתם מקבל "מזהה טבה". אם הוא לא קנה, האלגוריתם מקבל "קוזה רעה". ככל שהוא מוביל יותר מCONDOT טבות ורעות, האלגוריתם יודע לשפר את עצמו, לחתת משלך יותר לפרמטרים טובים ולהזינח פרמטרים פחות משמעותיים. אבל רגע, מי אמר למערכת להסתכל בכל סרטוני **Unboxing**?

האמת שהיא שף אחד. מישחו, בנאדים, אמר למערכת לזהות את כל הסרטונים שימוש צופה בהם ביטויוב, לזהות מתוך הסרטון, האודי, תיאור הסרטון ומילוט המפתח וכו' – איזה סוג סרטון זה. יתכן שאחרי מילאי-ידי ציבוריות הסרטונים, האלגוריתם מצליח למצוא קשר בין סוג מסוים של סרטונים לבין פעולות כמו רכישה באטר. באופן זהה, גוגל מזינה את האלגוריתם בכל הפעולות שהמשתמש מבצע. המילויים שהוא קוא, המזומות שהוא מסתובב בהם, התמצאות שהוא מעלה לעין, הבדיקות שהוא שוויה, כל מידע שיש אליו גישה. הכל שפרק לתרם מגאר התונאים העזום בו מנסה גוגל לבנות פרופילים ולמצוא קשר בין הטיסוי שלו לרכוש או כל פעולה אחרת שבאה לה לזהות.

המכונה המופלאה זו לומדת כל הזמן דברים חדשים ונונסה כל הזמן למצוא הקשות, לחזות תוצאות, לדקן היא הצלחה, ואם לא לתקן את עצמה שוב ושוב עד שהיא פוגעת במטריה. חשוב לציין שלמכונה אין סנטימנטים, כל המידע קביל ואם היא תמצא קשר מוכח בין מידת הנעלים של בנאדים לבין בייבי שארק, אז היא תשתמש בו גם אם זה לא נשמע הגיוני.

חשוב לשים לב לעניין המטריה – המטריה היא לא המצאה של האלגוריתם. הוא לא קם בבודק ומחליט מה האפליקציה שלם צריך לעשות. המטריה מוגדרת על ידי היוצר של המערכת. למשל – חישוב זמן סיוון, בניית מסלול אוטומטי בין A ל-B וכו'. המטריה של גוגל – משתמש בצע רכישה, והכל מתנקז להה בסוף, כי גול בראש ובראשונה היא מערכת פרטום. אגב, גם ההגדירה של מסלול "אוטומטי" היא מעשה ידי אדם. המכונה לא יודעת מה זה אוטומטי, זו רק מילה. אז צריך לעזרו לה ולהגיד לה שאוטומטי זה מונחים דם, מעס עזרות, כמה שפותות רמות וגו'. לסיום, המטריה מאופיינת על ידי האדם ולא על ידי המכונה. המכונה רק חותרת למטריה שהוגדר לה.

יש מנגנוני ML המתבססים על דатаה מסוודר ומתחוא כמו Netflix, עם כל המאפיינים של הסרטים אבל גם עם המאפיינים של הצופים (מדינה, גיל, שעת צפייה וכו'). לעומת זאת יש מנגנוני ML שמקבלים טיפה יותר חוש ומתבססים על מידע חלקית מאד (ש להם מידע על כל הסרטים, אבל אין להם מידע על ההצעה). מנגנונים אלו לא בהכרח מנגנים לבנות מונע המלצות אלא מנסים למצוא חוקיות בנתונים, ריגיות וכו'.

כך או כך, המערכת הסבור זהה הקורי **ML** בני מאלגוריתמים שונים המיפויים בינו לבין טקסט, אלגוריתמים אחרים המתמקדים בעיבוד אודיו, כללה המנתחים היסטוריית גליהא או זיהוי מותך דף ה-Web בו אתה צופים ועוד. عشرות או מאות מנגנונים כאלה מסתובבים ורצים ובונים את המפה השלמה. ככה רוב רשותה הפרטום הגדלוות שעבדות. ככל שהמכונה של גוגל/פיבובק תהיה חכמה יותר, ככה היא מடע להציג את המודעה המתאימה למשתמש הנכו, בזמן הנכו ועל ה-device המתאים.

1.1.2 Data, Tasks and Learning

כאמור, המטריה הבסיסית של למידת מכונה היא יכולת להכליל מוחלט מתוך הניסיון, ולבצע משימות באופן מודיעין ככל הנition על דטה חדש שעדיין לא נצפה, על בסיס צבירת ניסיון מדatta קיימ. באופן כללי ניתן לדבר על שלושה סוגים של למידה:

למידה חונשית (supervised learning) – הדטה הקים הינו אוסף של דוגמאות, וכל דוגמא יש תווית (label). מטרת האלגוריתמים במרקבה זה היא לLOSEOG דוגמאות חדשות שלא נצפו בתהיל הימידה. באופן פורמלי, עבור דטה $\mathcal{D} \in \mathbb{R}^{n \times d}$ – labels $\mathcal{Y} \in \mathbb{R}^{1 \times d}$ – יש אוסף $\mathcal{X} \in \mathbb{R}^d$, ומchapshim את האלגוריתם שמציע את המיפוי $Y \rightarrow X$: g. בצהורה הטובה ביותר, קלינור בהינתן דוגמא חדשה $x \in \mathcal{X}$, המכונה היא למצו עבורה את ה- y הנכו. המיפוי נמדד ביחס לפונקציות מהו, כפי שסביר בהמשך בוגע לתהיל הימידה.

למידה לא מונחית (unsupervised learning) – הדטה הקים הינו אוסף של דוגמאות מרוחב, בלי שנთן עליין מידע כלשהו המבחן בינהן. במקרה זה, בדרך כלל האלגוריתמים יחשו מודל המסביר את התפלגות הנקודות – למשל חלוקה לקבוצות שונות וצדומה.

למידה באמצעות חיזוקים (reinforcement learning) – הדטה בו נעזרים אינו מצוי בתחלת התוכנית אלא נאסף עם הזמן. ישם סוכנים הנמצאים בסביבה מסוימת ומעבירים מידע למשתמש, והוא בתורו למד אסטרטגיה בה הסוכנים ינקטו בצדדים הטוביים עבורה.

האלגוריתמים השונים של הלמידה מתחלקים לשתי קבוצות – מודלים דיסקרטוניים המוציאים פלט על בסיס מידע נתון, אך לא יכולים ליזור מידע חדש בעצמו, ומודלים גנרטיביים, שלא רק לומדים להכליל את הדעתה הלאמת גם עברו דוגמאות חדשות, אלא יכולים גם להבין את מה שהן רואו וליזור מידע חדש על בסיס הדוגמאות שנלמדו.

כאמור, בשבי לבנות מודל יש צורך בדעתה. מודל טוב הוא מודל שמסוגל להכליל מהדעתה הקיימת גם לדעתה חדשה. המודל למעשה מנסה למציא דפוסים בדעתה והקווים, ומהו הוא יכול להסביר סוכנות גם על דוגמאות חדשות. כדי לוודא שהמודל אכן מצליח להכליל גם על דוגמאות חדשות, בדרך כלל מחלקים את הדעתה הקיימת לשניים – קבוצת אימון (training set) וקבוצת מבחן (validation set). טס האימון מאפשר לסתור מודל להצלחת המודל – אם המודל מצליח למציא דפוסים בסיס האימון שנכונים גם עבור טס המבחן, זה סימן שהמודל הצליח למציא כלים שיכולים להיות נכונים גם לדוגמאות חדשות שבאו. לעיתים טס האימון מוחזקת בעצמה לשניים – קבוצת דוגמאות עליון המודל מתאימים, וקבוצת ולידיה (validation set) המשמשת להימנע מ-overfitting (validation set).

מגון התחומיים בהם משתמשים בכלים של למידה הוא עצום. עד כדי כך שIALIZED מציין באלגוריתמים לומדים. דוגמאות בולטות למשימות בהם משתמשים באלגוריתמים לומדים: סיוע, גרסה (מציאות קשור בין משתנים), חילוקה לקבוצות, מערכות המלצות, הזרת מד, ראייה ממוחשבת, עיבוד שפה טבעית ועוד.

1.2 Applied Math

האלגוריתמים של למידה מכונה נסמכים בעיקורם על שלושה ענפים מתמטיים; אלגברת ליניארית, חישוב דיפרנציאלי וסתירות. בפרק זה נציג את העקרונות הנדרשים בלבד, ללא הרחבה, על מנת להבין את הנושאים הנדרשים בספר זה.

1.2.1 Linear Algebra

קוטורים ומרחבים וקטוריים

באופן מתמטי מופשט, וקטורים, המנסונים בדרך כלל ע"י \vec{x} או על ידי x , הינם אובייקטים הנמצאים במרחב וקטורי $(+, \cdot)$ מעל שדה \mathbb{F} . מהו אותו מרחב וקטורי?

ראשית, השדה \mathbb{F} , הוא קבוצת מספרים המקיימים תכונות מתמטיות מסוימות. לדין בספר זה, השדה הוא קבוצת המספרים המשיים – \mathbb{R} , או קבוצת המספרים המרוכבים – \mathbb{C} . שנית, נשים לב כי המרחב הווקטורי דרוש גם הגדרת פעולת חיבור $(+)$.

כעת, $(+, +)$ היא מרחב וקטורי אם הוא מקיים את התכונות הבאות:

$$(I) \quad \text{קיים איבר אפס (וקטור אפס) } \vec{0} \text{ אשר } \vec{x} + \vec{0} = \vec{x} \quad (II) \quad \text{לכל איבר בשדה } a \text{ ולכל } \vec{x} \text{ ו-} \vec{y} \text{ בקוצה } V, \text{ גם } \vec{y} + a \cdot \vec{x} \text{ הינו איבר בקוצה } V.$$

הערה: קיימות דרישות נוספות למרחב וקטורי, אך הן מעבר לדריש בספר זה.

דוגמאות:

א. וקטורים גאומטריים:

מערך חד ממדי (x_1, x_2, \dots, x_n) (n -יה סדרה) נקרא וקטור גאומטרי a ממד', כאשררכיביו הוווקטור הם איברים בשדה \mathbb{F} . האיבר i , המיצג על ידי האידקס i מטא את מקום האיבר. מרחב זה מסומן ע"י \mathbb{F}^n .

נראה שמרחב זה הוא אכן מרחב וקטורי:

חיבור וקטורי:

$$\vec{x} = (x_1, x_2, \dots, x_n), \quad \vec{y} = (y_1, y_2, \dots, y_n) \rightarrow \vec{x} + \vec{y} = (x_1 + y_1, x_2 + y_2, \dots, x_n + y_n)$$

וקטור אפס:

$$\vec{0} = (0, 0, \dots, 0)$$

כפל בסקלר:

$$\vec{x} = (x_1, x_2, \dots, x_n) \rightarrow a \vec{x} = (a x_1, a x_2, \dots, a x_n)$$

הערה: לשם פשוטות, בהמשך, נenna וקטור גאומטרי כ"וקטור" בלבד.

ב. מטריצות:

מערך זו מגדיר, אשר רכיביו הם איברים בשדה \mathbb{F} , נקרא מטריצה מסדר $m \times n$, כאשר n הוא מספר השורות ו- m הוא מספר העמודות במערך. האיברים במטריצה A_{ij} מייצגים ע"י שני אינדקסים – i, j , המתארים את השורה והעמודה בהתאם. מרחב זה מסומן בדרך כלל ע"י $\mathbb{F}^{n \times m}$. נוכחות שמרחוב זה הוא אכן מרחב וקטורי:

חיבור מטריצות:

$$\hat{A} = \begin{pmatrix} A_{11} & \dots & A_{1m} \\ \vdots & \ddots & \vdots \\ A_{n1} & \dots & A_{nm} \end{pmatrix}, \hat{B} = \begin{pmatrix} B_{11} & \dots & B_{1m} \\ \vdots & \ddots & \vdots \\ B_{n1} & \dots & B_{nm} \end{pmatrix} \rightarrow \hat{A} + \hat{B} = \begin{pmatrix} A_{11} + B_{11} & \dots & A_{1m} + B_{1m} \\ \vdots & \ddots & \vdots \\ A_{n1} + B_{n1} & \dots & A_{nm} + B_{nm} \end{pmatrix}$$

מטריצת אפס:

$$\hat{0} = \begin{pmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{pmatrix}$$

כפל בסקלר:

$$\hat{A} = \begin{pmatrix} A_{11} & \dots & A_{1m} \\ \vdots & \ddots & \vdots \\ A_{n1} & \dots & A_{nm} \end{pmatrix} \rightarrow a \hat{A} = \begin{pmatrix} a A_{11} & \dots & a A_{1m} \\ \vdots & \ddots & \vdots \\ a A_{n1} & \dots & a A_{nm} \end{pmatrix}$$

ניתן להבחין כי הוקטורים הגיאומטריים שהוגדרו בדוגמה א', הם בעצם מטריצות בממד $1 \times n$.

ג. פולינומים:

פולינומים מסדר n הינם ביטויים מהסוג $a_0 + a_1x + a_2x^2 + \dots + a_nx^n$, כאשר n מייצג את החזקה הגדולה ביותר – a_i הם איברים בשדה. מרחב זה מסומן בדרך כלל ע"י $P_n(x)$.

בכל הדוגמאות לעיל קל להראות שהן אכן מהוות מרחב וקטורי. רשיימה חלקית לדוגמאות נוספות לווקטורים (ולמרחבים וקטוריים) כוללת למשל מרחב פונקציות או אפילו אוטונומ אלגברומנטים. כאן בחרנו להציג רק את הדוגמאות הרלוונטיות לשפר זה.

פעולות חשבון על מטריצות וקטוריים:

כמו שנציר לעיל, הוקטורים הגיאומטריים שהוגדרו בדוגמה א', הם בעצם מטריצות בממד $1 \times n = n$. לכן, פעולות החישוב מוגדרות באופן זהה.

• חיבור וחיסור בין שתי מטריצות:

$\hat{A} \in \mathbb{F}^{n \times k}$ כאשר A_{ij}, B_{ij} הם האיברים בשורה j בעמודה i של המטריצות \hat{A}, \hat{B} בהתחלה. אז, האיבר בשורה i בעמודה j של מטריצת הסכום (או ההפרש) הינו

$$(A \pm B)_{ij} = A_{ij} \pm B_{ij}$$

(הأدמת חיבור המטריצות בעצם כבר ניתנה בדוגמה א' לעיל)

שים לב: ניתן לחבר ולחסור מטריצות רק בעלות אותו הממד.

• כפל בין שתי מטריצות:

$\hat{A} \in \mathbb{F}^{k \times m}, \hat{B} \in \mathbb{F}^{m \times l}$ הן שתי מטריצות, אשר מסדר העמודות במטריצה \hat{A} שווה למספר השורות של מטריצה \hat{B} אך שתי המטריצות אינן בהכרח בעלות אותו ממד. במקרה כזה, מכפלת המטריצות מוגדרת על ידי:

$$\hat{A} \cdot \hat{B} = \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} + \dots + A_{1k}B_{k1} & \dots & A_{11}B_{1n} + \dots + A_{1k}B_{kn} \\ \vdots & \ddots & \vdots \\ A_{m1}B_{11} + \dots + A_{mk}B_{k1} & \dots & A_{m1}B_{1m} + \dots + A_{nk}B_{km} \end{pmatrix}$$

למעשה כל איבר בתוצאה הינו סכום של מכפלת שורה i ממטריצה A בעמודה j ממטריצה B :

$$(\hat{A} \cdot \hat{B})_{ij} = \sum_r A_{ir} B_{rj}$$

שים לב: על מנת שכפל המטריצות יהיה מוגדר מספר העמודות ב- \hat{A} שווה למספר השורות ב- \hat{B} .

עבור מטריצות ריבועיות (סדר $n \times n$), מוגדר גם הכפל $\hat{A}\hat{B}$ וגם ההפוך $\hat{B}\hat{A}$, אולם יתכן ש- $\hat{A}\hat{B} \neq \hat{B}\hat{A}$.

• שחילוף (transpose):

החלפת שורות בעמודות, או 'סיבוב' המטריצה. נניח מטריצה $\hat{A} \in \mathbb{F}^{n \times m}$, אז השחלוף שלה, המסומן כ- \hat{A}^T הוא:

$$(\hat{A}^T)_{ij} = A_{ji}$$

ובאופן מפורש:

$$\hat{A} = \begin{pmatrix} A_{11} & \cdots & A_{1m} \\ \vdots & \ddots & \vdots \\ A_{n1} & \cdots & A_{nm} \end{pmatrix} \rightarrow \hat{A}^T = \begin{pmatrix} A_{11} & \cdots & A_{n1} \\ \vdots & \ddots & \vdots \\ A_{1m} & \cdots & A_{nm} \end{pmatrix}$$

שים לב שהמטריצה החדשה \hat{A}^T הינה במד $n \times m$. בנוסף ניתן להוכיח כי מתקיימים:

שחלוף של וקטור שורה, נตอน וקטור عمودה ולהפך.

• מטריצת יחידה:

מטריצת יחידה, הינה מטריצה ריבועית (סדר $n \times n$), המסומנת על ידי \mathbb{I}_n ומוגדרת כך שכל איבריה מלבד איברי האלכסון הראשי המקבילים את הערך 1:

$$(\mathbb{I}_n)_{ij} = \begin{cases} 1 & i=j \\ 0 & i \neq j \end{cases}$$

ובאופן מפורש:

$$\mathbb{I}_n = \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{pmatrix}$$

מטריצה זו מקיימת $\mathbb{I}_m \cdot \hat{A} = \hat{A} = \hat{A} \cdot \mathbb{I}_n$ מסדר $m \times n$.

הערה: לעיתים סדר מטריצת היחידה אינו משנה או טריוויאלי, ולכן המטריצה מסומנת רק על ידי \mathbb{I} ללא ציון הממד.

• מטריצה הופכית:

למטריצות ריבועיות (מטריצות עם מספר זהה של שורות ועמודות; סדר $n \times n$) יתכן שיש מטריצה הופכית A^{-1} שמקיימת את הקשר:

$$\hat{A} \cdot \hat{A}^{-1} = \hat{A}^{-1} \cdot \hat{A} = \mathbb{I}_n$$

דוגמא: $\hat{A} = \hat{A}^{-1} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. לכן, במקרה זה $(\hat{A}^{-1})_{ij} = (\hat{A})_{ji}$.

• מטריצה צמודה/הרמייטית:

עבור מטריצה A , המטריצה $A^* = A^\dagger$ נקראת הצמוד הרמייטי של A , ומתקיימים:

$$(A^*)_{ij} = \overline{A_{ji}}$$

הצמוד הרמייטי הוא שחלוף של A , כאשר לכל איבר במטריצה המשוחלפת לוקחים את הצמוד המרוכב.

אם A מטריצה ממשית, המטריצה הצמודה שלה היא למעשה המטריצה המשוחלפת של A .

• **מטריצה אוניטרית:**

מטריצה אוניטרית היא מטריצה ריבועית מעל המספרים המורכבים המקיימת את התנאי:

$$A^*A = AA^* = \mathbb{I}$$

מערכת משוואות לינאריות:

מערכת משוואות לינארית מוצגת באופן כללי באופן הבא:

$$\begin{array}{l} A_{11}x_1 + A_{12}x_2 + \dots + A_{1n}x_n = b_1 \\ \vdots \\ A_{m1}x_1 + A_{m2}x_2 + \dots + A_{mn}x_n = b_m \end{array}$$

נשים לב כי מערכת משוואות לינארית ניתן לייצוג באופן קומפקטי על ידי הפרדה בין רשימת המשתנים, המקדמים של משתנה, והאיבר החופשי, באופן הבא:

$$\hat{A} \vec{x} = \vec{b} = \begin{pmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{m1} & \cdots & A_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$

מטריצה \hat{A} הינה מטריצה המקדמים מסדר $m \times n$, כאשר n הוא מספר המשתנים, ו- m הוא מספר המשוואות במערכת.

$$\text{קטgor} \vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \text{הינו וקטור عمودה (לעתים גם מסומן על ידי } (x_1, x_2, \dots, x_n)^T \text{), המיצג את וקטור המשתנים.}$$

$$\text{קטgor} \vec{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \text{הינו וקטור عمودה, שיאברו הם האיבר החופשי.}$$

פתרונות של מערכת המשוואות הלינארית, $\vec{b} = \hat{A} \vec{x}$, אם הם קיימים ויחדים, נתונים ע"י $\vec{x} = \hat{A}^{-1} \vec{b}$ (בהנחה והמטריצה ריבועית).

מכפלה פנימית, נורמה, אורותוגונליות

מרחיב מכפלה פנימית, מוגדר על ידי מרחב וקטורי V (המוגדר על גבי שדה \mathbb{F}) ועל ידי פעולה "מכפלה פנימית". מכפלה פנימית, המקרה לעתים רק מכפלה, הינה בעצם פונקציה המקבלת שני וקטורים ממוחזר וקטורי V ומחזירה סקלר (=מספר) בשדה \mathbb{F} . מכפלה זו, מסומנת בדרך כלל ע"י $\langle \cdot, \cdot \rangle$ (או ע"י $\mathbb{F} : V \times V \rightarrow \mathbb{F}$: $\langle \cdot, \cdot \rangle$), חייבת לכך מספר תכונות.

לכל $V \in \mathbb{F}$, $\vec{v}, \vec{u}, \vec{w} \in V$ (כל שלושה וקטורים במרחב הווקטורי V), ולכל $\mathbb{F} \in \lambda$ (סקלר בשדה \mathbb{F}):

$$\begin{aligned} \langle \vec{v} + \vec{u}, \vec{w} \rangle &= \langle \vec{v}, \vec{w} \rangle + \langle \vec{u}, \vec{w} \rangle && \bullet \\ \langle \lambda \vec{v}, \vec{u} \rangle &= \lambda \langle \vec{v}, \vec{u} \rangle && \bullet \\ \langle \vec{v}, \vec{u} \rangle &= \langle \vec{u}, \vec{v} \rangle && \bullet \\ \langle \vec{v}, \vec{v} \rangle &\geq 0 && \bullet \end{aligned}$$

ההגדירה עצמה של המכפלה משתנה כתלות במרחב הווקטורי הנתון. לדוגמה:

A. מכפלה סקלרית על מרחב הווקטורים הגיאומטריים:

נתונים $\vec{v}, \vec{u} \in \mathbb{C}^n$ וקטורים גיאומטריים מסדר n מעל שדה המספרים המורכבים. מכפלה פנימית בין שני וקטורים אלו, נראית גם מכפלה סקלרית, המוגדרת על י"ד:

$$\langle \vec{v}, \vec{u} \rangle = \vec{v}^T \cdot \vec{u} = \sum_{i=1}^n \bar{v}_i u_i$$

כאשר \vec{v} הינו הצמוד המרוכב של v .

ב. מרחב הליברט – מרחב מכפלה פנימית על מרחב הפונקציות:

נניח שתי פונקציות מרוכבות $f, g : \mathbb{C} \rightarrow \mathbb{C}$: אינטגרביליות בתחום כלשהו / (כמו שהוזכר לעיל, גם מרחב הפונקציות הוא מרחב וקטורי), אז המכפלה פנימית מוגדרת על ידי:

$$\langle f(x), g(x) \rangle = \int_I f^*(x)g(x)dx$$

כאשר $(x)^*$ הינו הצמוד המרוכב של $f(x)$.

ניתן להגדיר גם מרחבי מכפלה פנימית נוספים, נניח עבור מרחב המטריצות.

נורמה:

נורמה, מוגדרת על ידי מכפלה פנימית של וקטור בעצמו, וסומנת " $\|\cdot\|$ ", זאת אומרת:

$$\|\vec{v}\| = \sqrt{\langle \vec{v}, \vec{v} \rangle} \geq 0$$

שווין מתקיים אך ורק עבור וקטור האפס; $\vec{v} = 0 \Leftrightarrow \|\vec{v}\| = 0$.

תכונה נוספת, נקראת אי-שוויון המשולש, מתחזרת על ידי:

$$\|\vec{u} + \vec{v}\| \leq \|\vec{u}\| + \|\vec{v}\|$$

אי-שוויון נוסף הקשור לנורמות נקרא אי-שוויון קושי-שוורץ (Cauchy-Schwarz inequality)

$$\|\vec{u}\| \cdot \|\vec{x}\| \leq |\langle \vec{u}, \vec{x} \rangle|$$

כאשר $\langle \vec{u}, \vec{x} \rangle$ הינה המכפלה הפנימית בין שני וקטורים, המוגדרת מעל הטעמים $\vec{x} = \sum_i x_i \vec{e}_i$, $\vec{u} = u \cdot \vec{e}_i$, והביטוי $\|\vec{u}\| \cdot \|\vec{x}\|$ הוא מכפלת הנורמות.

דוגמאות:

א. במרחב הווקטוריים הגיאומטריים, הגדרת הנורמה היא בעצם הגדרת אורך (או גודל הווקטור). נניח עבור הווקטורים הגיאומטריים התלת-ממדים, $V = \mathbb{R}^3$, אץ עבור $\vec{V} \in V$ ($\vec{V} = \vec{v}$, הנורמה מוגדרת " $\|\vec{v}\|$ ".

ב. במרחב הליברט נורמה של פונקציה $f : \mathbb{C} \rightarrow \mathbb{C}$ הינה $\|f\|^2 = \int_I |f(x)|^2 dx$.

אורותוגונליות

הגדרת מכפלה פנימית מאפשרת לנו להגיד אורתוגונליות (או אנכיות) של שני וקטורים במרחב מכפלה פנימית מסוים. שני וקטורים $\vec{u}, \vec{v} \in V$ נקראים אורתוגונליים זה לזה אם ורק אם המכפלה הפנימית שלהם הינה אפס:

$$\langle \vec{u}, \vec{v} \rangle = 0 \Leftrightarrow \vec{u} \perp \vec{v}$$

כאשר מתייחסים למרחב הווקטוריים הגיאומטריים, קל להבין את משמעות האורתוגונליות.

אורותוגונליות היא הכללה של תכונות הניצבות המוכרת מגאומטריה. בגאומטריה, שני ישרים במרחב האוקלידי ניצבים זה לזו אם חזיתות הנוצרת בנקודת החיתוך שלהם היא חזית ישרה (בת 90 מעלות). משג האורתוגונליות מכליל תכונה זו גם למרחבים ווקטוריים n -ממדים. על מנת להכליל את מושג הניצבות יש ראייה לכך חזית בין שני וקטורים:

$$\cos(\theta) = \frac{\langle \vec{u}, \vec{v} \rangle}{\|\vec{u}\| \cdot \|\vec{v}\|}$$

לפי אי-שוויון קושי-שוורץ מתקיים: $|\langle \vec{u}, \vec{v} \rangle| \leq \|\vec{u}\| \cdot \|\vec{v}\|$, ולכן הביטוי באך ימי תמיד קטן או שווה בערכו המוחלט – 1. כיוון שכן, תמיד ניתן לחשב חזית בין שני וקטורים בעדרת מכפלה פנימית.

לוקטוריים אורתוגונליים חשובות רביה כאשר חוקרים מרחבים וקטוריים. לבסיס של מרחב וקטורי יש מספר תוכנות נוחות כאשר הוא אורתונורמלי (אל אברי אורתוגונליים זה לזה ובועל אוור 1). יתר על כן, מתרברר שהণית בסיס כלשהו למרחב וקטורי ניתן לקבל ממנה בסיס חדש של אל אברי אורתוגונליים זה לזה, כך שתמיד ניתן למצאו בסיס נוח שכזה. דבר זה נעשה על ידי תהליך גרט-שmidt (gram-schmidt).

שי וקטוריים אורתוגונליים יסומנו על ידי \perp . עבור וקטוריים אורתוגונליים מתקיימות התכונות הבאות:

- אם $v \perp u$, אז $v \perp w$.
- אם $v \perp u$, אז לכל סקלר λ גם $v \perp \lambda u$.
- אם $v \perp u$ וגם $v \perp w$, אז $v \perp (u + w)$.
- אם וקטור אורתוגונלי לקבוצה של וקטוריים אזי הוא גם אורתוגונלי לכל צירוף לינארי שלהם (נובע משתי התכונות הקודומות).

וקטוריים עצמיים וערכים עצמיים

תהי $n \times n$ מטריצה ריבועית, וקטור $v \in \mathbb{F}^n$ הוא סקלר. ג נקרא ערך עצמי של A אם $v \neq 0$ ונראה הוקטור העצמי המתאים אם מתקיים:

$$A \cdot v = \lambda \cdot v$$

ניתן להראות שעבור מטריצה A , הוקטוריים העצמיים המתאימים לסקלר λ הם כל פתרונות המשוואה ההומוגנית $(A - \lambda I_n) \cdot v = 0$.

אם נסמן $[v_n, \dots, v_1] = [\lambda_1, \dots, \lambda_n, V]$ אז מתקיים:

$$A = V \operatorname{diag}(\Lambda) V^{-1}$$

כאשר (Λ) הוא ערכי האלכסון של המטריצה A .

פירוק לערכים סינגולריים

ניתן לפירק מטריצה $n \times m$ M למכפלה של שלוש מטריצות באופן הבא:

$$M = U \Sigma V^*$$

כאשר $m \times m$ U היא מטריצה אוניטרית מרכיבת (או ממשית), $n \times n$ V היא מטריצה אלכסונית שכל איברי האלכסון שלה ממשיים ואי-שליליים, ו- $n \times m$ Σ היא מטריצה אוניטרית מרכיבת (או ממשית). פירוק זה נקרא פירוק לערכים סינגולריים (Singular value decomposition - SVD).

ערך האלכסון של Σ_{ii} – מסודרים מהגדול לקטן, והם נקראים הערכים הסינגולריים של M . בנוסף, m העמודות של U נקראות וקטוריים הסינגולריים השמאליים של M , ובההתאמה n העמודות של V הן הוקטוריים הסינגולריים היוניים של M . שלוש המטריצות מקיימות את התכונות הבאות:

- הוקטוריים הסינגולריים השמאליים של M הם וקטוריים עצמיים של $M M^*$.
- הוקטוריים הסינגולריים הימניים של M הם וקטוריים עצמיים של $M^* M$.
- הערכים הסינגולריים (אי-בריאלי האלכסון של Σ) שאינם אפס הם שורשים ריבועיים של הערכים עצמיים השונים מאפס של $M M^*$ ושל $M^* M$.

לפירוק SVD יש שימושים בתחומיים רבים, ואף ניתן להציגו בעזרתו נורמות חדשות.

1.2.2 Calculus

פונקציה

פונקציה הינה התחמה (או העתקה), המתאימה לכל איבר x (בתחום מסוים), ערך ייחיד y , ומוסמנת באופן הבא: $f(x) = y$. קבוצת הא-ים, נקראת תחום, וקבוצת ה-ים נקראת טווח. קבוצות התחום והטווח יכולות להיות רציפות (למשל מספרים ממשיים חיוביים או בדידות (למשל קבוצה $\{0,1\}$). בדרך כלל הסימון מופיע כך: $f: X \rightarrow Y$, כאשר X ו- Y הינם התחום והטווח בהתחמה.

דוגמא: $\mathbb{R}^+ \rightarrow \mathbb{R}^2 = [\cdot | \cdot]$, הינה פונקציה, הולכת וקטורים גיאומטריים דו-ממדים, ומהזירה מספר ממשי אי שלילי.

הפונקציה עצמה $[\cdot | \cdot]$ היא הנורמה של הווקטור, כפי שהוגדרה בפרק הקודם.

נגזרת

עבור פונקציות ממשיות, נזרת מוגדרת על ידי מידת השתנות של הפונקציה $(x) f$ על ידי שינוי קטן (Δx) (אינפיניטסימלי) בא. באופן גיאומטרי, הנזרת הינה השיפוע של הפונקציה בנקודה a . נזרת מסומנת בדרך כלל ע"י $f'(x) = \frac{df}{dx}(x)$

נגזרות של פונקציות אלמנטריות ניתן לחשב באמצעות כלים ידועים. לדוגמה:

- לכל $0 \neq a$ מתקיים: $\frac{d(x^n)}{dx} = nx^{n-1}$
- חיבור או חיסור פונקציות: $\frac{d(f(x)+g(x))}{dx} = \frac{df(x)}{dx} + \frac{dg(x)}{dx}$
- מכפלת שתי פונקציות: $\frac{d(f(x) \cdot g(x))}{dx} = \frac{f(x)dg(x)}{dx} + \frac{g(x)df(x)}{dx}$
- כלל שרשרת: $\frac{df(g(x))}{dx} = \frac{df}{dg} \frac{dg}{dx}$

כיוון שנזרת של פונקציה ממשית מכמתת את קצב שינוי הפונקציה, אז בתחום שבו הפונקציה יורדת הנזרת שם תהיה שלילית, ובתחום שבו היא עולה הנזרת תהיה חיובית. ככל שקצב ההשתנות גדול יותר כך ערכיה המוחלט של הנזרת גדול.

הערה: לא לכל פונקציה מוגדרת נזרת. מספר זה נניח שהפונקציה אנליטית ולכך גזירה.

הערה נוספת: כיוון שנזרת של פונקציה היא גם פונקציה, ניתן גם להגדיר נזרת שנייה או נזרת מסדרים גבוהים יותר. בדרך כלל הסימון $\frac{d^2f}{dx^2}(x) = f''(x)$ לנזרת מסדר שני וכוכלי

נקודות אקסטרום

נקודות אקסטרום של פונקציה, הן נקודות שבהם הפונקציה מקבלת ערך מקסימום או מינימום באופן מקומי. בנקודות אלו, הנזרת של הפונקציה "משנה כיון" (פונקציה עולה לפונקציה יורדת או להפך) ולכן מקבלת את הערך אפס. יש לשים לב שההתפקידו הנזרת הביקורת המינימום והמקסימום היא תנאי הכרחי אך לא מספק. יתרון שהנזרת מתאפסת בנקודה מסוימת, אך נקודה זו אינה מינימום או מקסימום מקומי, אלא נקודת פיטול.

לדוגמה: $f(x) = x^3$. נזרת הפונקציה הנה $f'(x) = 3x^2$ והיא מתאפסת בנקודה $0 = x$.

גרדיינט, יעקוביאן והסיאן

עבור פונקציה מרובת משתנים, נזרת חלקית מוגדרת להיות הנזרת של הפונקציה לפי אחד המשתנים בלבד, והיא מסומנת ב- $\frac{\partial f}{\partial x_i}(x_1, \dots, x_n)$. כאשר גוזרים לפי משתנה מסוים, שאר המשתנים הם קבועים ביחס לנזרת. בהינתן הפונקציה $f(x_1, \dots, x_n)$, וקטור הנזרות לפי כל המשתנים נקרא גרדיינט:

$$\nabla f(x_1, \dots, x_n) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \leftrightarrow [\nabla f]_i = \frac{\partial f}{\partial x_i}$$

עבור n פונקציות התייחסות ב- n משתנים, הייעקוביאן הוא מטריצת הנזרות החלקיים:

$$\mathcal{J}_f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}_{n \times m} \leftrightarrow [\mathcal{J}_f]_{ij} = \frac{\partial f_i}{\partial x_j}$$

עבור פונקציה $f(x_1, \dots, x_n)$, מטריצת הנזרות מסדר שני נקראת הסיאן:

$$\mathcal{H}_f = \nabla^2 f = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}_{n \times n} \leftrightarrow [\mathcal{H}_f]_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$$

שני כללים חשובים בחישוב נגזרות של מטריצות:

$$\begin{aligned}\nabla_x(a^T x) &= a \\ \nabla_x(x^T A x) &= (A + A^T)x\end{aligned}$$

1.2.3 Probability

תורת הסתברות היא תחום המספק כלי ניתוח למאורעות המיליכים ממד של אקראיות ואינטראקטיביות. הסתברות של מאורע הוא ערך מסוימרי למדת הסתברות שהוא יתרחש, כאשר ערך זה נע בין 0 ל-1 – מאורע בלתי אפשרי הוא בעל הסתברות 0, ומאורע ודאי הוא בעל הסתברות 1.

הגדרות בסיסיות

Ω = מרחב המדגם – מכלול האפשרויות השונות של ניסוי. לדוגמה עבור הטלת קובייה: $\Omega = \{1, 2, 3, 4, 5, 6\}$.

$A = \{2, 4, 6\}$ = even number – קבוצה – חלק מרחב המדגם. לדוגמה עבור הטלת קובייה: A – קבוצה – חלק מרחב המדגם. שוויון $p(A) = \frac{\#\text{elements in } A}{\#\Omega}$.

מאורע – תוצאה אפשרית של ניסוי. הסתברות – סיכוי של מאורע להתרחש. עבור תוצאה A של מרחב המדגם Ω , ההסתברות ל'קיים מאורע מקבוצת A שווה לחלק היחסי של מספר איברי הקבוצה מתוך קבוצת המדגם:

$$p(A) = \frac{\#A}{\#\Omega}, \quad 0 \leq p(A) \leq 1$$

$A \cup B$ = איחוד – איחוד של שתי קבוצות הוא אוסף האברים של שתיהן. איחוד של הקבוצות A ו- B הוא אוסף האיברים המופיעים לפחות פעם אחת ב- A או B . לדוגמה עבור הטלת קובייה:

$$A = \text{even number} = \{2, 4, 6\}, \quad B = \text{lower than 4} = \{1, 2, 3\}$$

$$\rightarrow A \cup B = \{1, 2, 3, 4, 6\}, \quad p(A \cup B) = \frac{5}{6}$$

$A \cap B$ = חיתוך – חיתוך של שתי קבוצות הוא אוסף האיברים המופיעים בשתי הקבוצות. חיתוך של הקבוצות A ו- B הוא אוסף האיברים המופיעים גם ב- A וגם ב- B . עבור הדוגמא הקודמת:

$$A \cap B = \{2\}, \quad p(A \cap B) = \frac{1}{6}$$

מאורעות זרים – מאורעות שהחיתוך שלהם ריק, כלומר אין להם איברים משותפים:

$$A \cap B = \emptyset, \quad p(A \cap B) = 0$$

מאורע משלים – מאורע המכיל את כל האיברים שאינם נמצאים בקבוצה מסוימת:

$$A \cup A^c = \Omega \rightarrow p(A \cup A^c) = 1, \quad p(A) = 1 - p(A^c)$$

מאורעות בלתי תלויים: $P(A \cap B) = P(A) \cdot P(B)$. באופן אינטואיטיבי ניתן לחשב על כך שבמקרה כזה ידיעת אחד אינה משפיעה על הסיכוי של השני.

אם המאורעות זרים (וهم בעלי סיכוי שונה מ-0), הם בהכרח תלויים:

$$P(A \cap B) = 0 \neq P(A) \cdot P(B) > 0$$

$p(A|B)$ = הסתברות מותנית – בהינתן מידע מסוים, מה ההסתברות של מאורע כלשהו:

$$p(A|B) = \frac{p(A \cap B)}{p(B)} \leftrightarrow p(A|B) \cdot p(B) = p(A \cap B) = p(B|A) \cdot p(A)$$

בעזרת הגדרה של הסתברות מותנית ניתן לתת הגדרה נוספת למאורעות בלתי תלויים:

$$A, B \text{ תלויים} \leftrightarrow p(A|B) = p(A)$$

נשים לב שהמשמעות של שתי הגדרות זהה – המידע על B לא משנה את חישוב ההסתברות של A .

נוסחת ההסתברות השלמה וחוק ביוס

נוסחת ההסתברות השלמה היא נוסחה פשוטה המאפשרת לחשב מאורעות מסוימים. ניתן לפרק מרחב ההסתברות ל/events זרים, ואז לחשב את ההסתברות של כל אחד בפני עצמו. אם ניקח את כל ההסתברויות המתקבלות, ונכפיל כל אחת מהן במשקל של אותו איבר, נקבל את נוסחת ההסתברות השלמה:

$$P(B) = \sum_i P(B|A_i) \cdot P(A_i)$$

מתוך נוסחה זו מגאים בקלהות לחוק ביוס, המאפשרת לחשב הסתברות מותנית באמצעות ההטניה ההפוכה:

$$p(A|B) = \frac{p(A \cap B)}{p(B)} = \frac{p(B|A)p(A)}{p(B)}$$

משפט הכללה וההדחה

כדי לסייע עצמים בקבוצה, אפשר לכלול ולהוציא את אותו עצם שוב ושוב, כל עוד בסוף ההליך נספר כל עצם פעם אחת. עקרון פשוט זה מתורגם לנוסחה הבאה:

$$\left| \bigcup_{i=1}^n A_i \right| = \sum_{i=1}^n |A_i| - \sum_{1 \leq i < j \leq n} |A_i \cap A_j| + \sum_{1 \leq i < j < k \leq n} |A_i \cap A_j \cap A_k| - \dots + (-1)^{n-1} |A_1 \cap \dots \cap A_n|$$

עבור 2 קבוצות הנוסחה נהיה יותר פשוטה:

$$|A \cup B| = |A| + |B| - |A \cap B|$$

במקרה זה, כאשר A, B זורות, אז $|A \cap B| = 0$.

עבור שלוש קבוצות מתקבלת הנוסחה:

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |B \cap C| - |A \cap C| + |A \cap B \cap C|$$

משתנים אקראיים

ר' $\Omega \rightarrow \mathbb{R}$: משתנה מקרי – פונקציה המתאימה לכל מאורע השיר למרחב ההסתברות ערך מסווני, המהווה את סיסמי של המאווע להתרחש.

פונקציית ההסתברות של משתנה מקרי X נותנת את הסיסמי של כל x אפשרי:

$$f_X: \mathbb{R} \rightarrow [0,1] = p(X = x)$$

פונקציה זו מקיימת שלוש אקסiomות:

- הסתברות של כל מאורע במרחב המודגם גדולה או שווה ל-0.
- סכום ההסתברויות של כל המאורעות במרחב שווה ל-1: $\sum p(X = x) = p(\Omega) = 1$.
- סכום ההסתברויות של שני מאורעות זרים שווה להסתברות של איחוד המאורעות.

עבור משתנה מקרי רציף יש אינסוף מאורעות אפשריים, אך ההסתברות של כל מאורע יחיד היא 0. לכן עבור משתנה מקרי רציף מכילים את פונקציית ההסתברות לפונקציה הנקראת פונקציית ההתפלגות (או פונקציית הצפיפות המצטברת), המחשבת את ההסתברות שאירוע יהיה קטן מערך מסוים:

$$F_X(a) = p(X \leq a) = \int_{-\infty}^a f_X(x) dx$$

נינתן לחשב בעזרת פונקציה זו את ההסתברות שמאורע γ יהיה בטווח מסוים:

$$p(a \leq X \leq b) = F_X(b) - F_X(a) = \int_a^b f_X(x) dx$$

פונקציית ההתפלגות מקיימת את התכונות הבאות:

- $\lim_{a \rightarrow -\infty} F_X(a) = 0$
- $\lim_{a \rightarrow \infty} F_X(a) = 1$
- $\int_{-\infty}^{\infty} f_X(x)dx = 1$
- הפונקציה מונוטונית
 $\geq a$) = $1 - F_X(a)$
- הפונקציה גזירה ומשתנה

תכונות ופרמטרים עבור משטנה מקרי

תוחלת – מוצע משקל של כל הערכים האפשריים, כאשר כל ערך מוכפל בהסתברות שלו (במקרה היחיד – סכום במקרה הרצף – אינטגרל):

$$\mathbb{E}[X] = \sum_i x_i P(X = x_i) / \int_{-\infty}^{\infty} xf(x)dx$$

תכונות:

$\mathbb{E}[c] = c$
 $\mathbb{E}[\mathbb{E}[X]] = \mathbb{E}[X]$
 לינאריות התוחלת:

שונות – מدد פיזור הערכים ביחס למרכז המשוקל (-התוחלת):

$$Var[x] = E[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2 = \int_{-\infty}^{\infty} x^2 f(x) dx - (\mathbb{E}[X])^2$$

סטיית תקן מוגדרת להיות שורש השונות: $\sigma = \sqrt{Var[X]}$

תכונות:

$$\text{Var}[x] \geq 0$$

שונות משותפת – מدد ליחס אפשרי בין שני משתנים מקרים:

$$cov(X, Y) = \mathbb{E}[X \cdot Y] - \mathbb{E}[X] \cdot \mathbb{E}[Y]$$

כאשר: $\mathbb{E}[X \cdot Y] = \sum_j \sum_i x_i y_j P(X = x_i \cap Y = y_j)$. אם המשתנים בלתי תלויים אז מתקיים 0

מקדם המתאים – נרמול של השונות המשותפת: $\rho(X, Y) = \frac{\text{cov}(X, Y)}{\sqrt{V(X)V(Y)}}$.

שני משתנים מקרים מוגדרים בלתי מתואמים אם $0 = \text{cov}(Y, X)$. אם המשתנים בלתי תלויים אז הם בהכרח בלתי מתואמים.

. $V[X + Y] = V[X] + V[Y] + 2 \cdot cov(X, Y)$: ניתן לכתוב

פונקציה יוצרת מומנטים (התמרת לפולס של פונקציית הצפיפות – $\{t\}$)

$$M_X(t) = \mathbb{E}[e^{tX}] = \int_{-\infty}^{\infty} e^{tx_i} p_X(x_i) dx$$

בעזרת פונקציה זו ניתן ליצור מומנטים, שימושיים ללמידה על המשתנים:

$$\frac{d^n M_X(t)}{dt^n} \Big|_{t=0} = \mathbb{E}[X^n]$$

המומנט הראשון הוא התוחלת והמומנט השני הוא כפעת זהה לשונות (מומנט השני הוא $M''(0) = E[X^2]$, לעומת השונות שהיא $(Var(X) = E[X^2] - E[X]^2)$).

התפלגות מיוחדות (בדיד)

ישנן כל מיני התפלגותים מיוחדות, שימושיות בטבע בכל מיני מקרים ויש להן נוסחאות ידועות.

התפלגות ברנולי: $X \sim Ber(p)$

ניסוי בעל שתי תוצאות אפשריות "הצלחה" או "כשלון". המשתנה המקרי מקבל שני ערכים בלבד – 0 או 1, בהתאם להצלחה וכישלון.

$$P(X = k) = \begin{cases} 1, & k = 1 \\ 0, & k = 0 \end{cases}, \mathbb{E}[X] = p, V[X] = pq = p(1-p)$$

התפלגות בינומית: $X \sim B(n, p)$

בהתפלגות בינומית חוזרים על אותו ניסוי ברנולי n פעמים באופן בלתי תלוי זה בזה. מגדירים את X להיות מספר ההצלחות שהתקבלו בסה"כ. סמן k כמספר ההצלחה בניסוי בודד ו- q כמספר בניסוי בודד.

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}, \mathbb{E}[X] = np, Var[X] = npq$$

צריך לודא 3 דברים: 1) חוזרים על אותו ניסוי באופן בלתי תלוי. 2) חוזרים על הניסוי n פעמים. 3) X מוגדר כמספר ההצלחות המתקבלות בסה"כ.

התפלגות גיאומטרית: $X \sim G(p)$

חוזרים על ניסוי ברנולי. כאשר X מבטא את מספר הניסויים שבוצעו עד ההצלחה הראשונה. k מסמן את הסתרבות ההצלחה בניסוי בודד.

$$P(X = k) = pq^{k-1}, \mathbb{E}[X] = \frac{1}{p}, V[X] = \frac{q}{p^2}$$

בהתפלגות זו יש שתי תכונות נוספות:

$$(1) \text{ "תיכון חוסר זיכרון": } P[X = (n+k)] = P(n)$$

$$(2) \text{ ההסתברות שיעברו } k \text{ ניסויים ללא הצלחה: } P[X > k] = q^k$$

כמו כן, אם מעוניינים לדעת את מספר הניסיונות הממוצע הנדרש עד להצלחה ראשונה – יש לחשב את התוחלת של המשתנה המקרי X .

התפלגות אחידה: $X \sim U[a, b]$

בהתפלגות זו לכל תוצאה יש את אותה הסתברות. הערכים המתקבלים בהתפלגויות החל מ- a ועד b הינם בקפיצות של יחידה אחת (לדוגמא הגרלה של מספר שלם בין 1-100):

$$P(X = k) = \frac{1}{b-a+1}, k = a, a+1, \dots, b, \mathbb{E}[X] = \frac{a+b}{2}, V[X] = \frac{(b-a+1)^2 - 1}{12}$$

התפלגות פואסונית: $X \sim poi(\lambda)$

התפלגות זאת מתאפיינת במספר אירועים ליחידת זמן כאשר λ הוא פרמטר המיצג את קצב האירועים ליחידת זמן הנברשת.

$$P(X = k) = \frac{e^{-\lambda} \lambda^k}{k!}, k = 1 \dots \infty, \mathbb{E}[X] = Var[X] = \lambda$$

יש לשים לב שכאן ההתפלגות נזدة ליחידת זמן.

התפלגות היפר גאומטרית: $X \sim H(N, D, n)$

נתונה אוכלוסייה שמכילה N פריטים סה"כ, מתייחסים "מיוחדים" בעלי תכונה מסוימת. בוחרים ממנה אוכלוסייה n פריטים ללא החזרה. מגדירים את X להיות מספר הפריטים ה-"מיוחדים" שבדגמו.

$$P(X = k) = \frac{\binom{D}{k} \binom{N-D}{n-k}}{\binom{N}{n}}, \mathbb{E}[X] = \frac{nD}{N}, Var[X] = \frac{nD}{N} \left(1 - \frac{D}{N}\right) \frac{N-n}{N-1}$$

התפלגות בינומית שלילית: $X \sim NB(r, p)$

חווריים על אותו ניסוי ברולוי בדקה אחר זה באופן בלתי תלוי עד אשר מצליחים בפעם ה- r . לעומת זאת, מבצעים את הניסוי עד שמבצעים r פעמים. מגדירים את X להיות מספר החזרות עד שהתקבלו r הצלחות.

$$P(X = k) = \binom{k-1}{r-1} p^r (1-p)^{k-r}, k = r, r+1, \dots, \infty, \mathbb{E}[X] = \frac{r}{p}, Var[X] = \frac{r(1-p)}{p^2}$$

התפליגיות מיוחדות (רכיף)

התפליגות מעריכית: $X \sim exp(\lambda)$

התפליגות רציפה המאפיינת את הזמן עד להתרחשות מאורע מסוים. λ הוא ממוצע מספר האירועים המתהרחשים ביחידת זמן (אותו פרמטר מההתפלגות הפואסונית). $\lambda < 0, X \sim exp(\lambda)$.

אם בהתפלגות זו יש את תכונות חוסר הדיזכרון: $P(X > (a+b)|X > a) = P(x > b)$

התפליגות אחידה: $X \sim U(a, b)$

זו התפליגות שפונקציית הצפיפות שלה קבועה בין a ל- b .

$$f(x) = \frac{1}{b-a}, a < x < b, f(x) = 0 \text{ אחרת.}$$

$$\mathbb{E}[X] = \frac{a+b}{2}, Var[X] = \frac{(b-a)^2}{12}$$

התפליגות נורמלית: $X \sim \mathcal{N}(\mu, \sigma^2)$

התפליגות נורמלית היא התפליגות חשובה מאוד כיוון שהיא מופיעה בהמוני מקרים. פונקציית הצפיפות של ההתפלגות הנורמלית נראה כmo פעמון, כאשר לעוקמה קוראים גם עיקומת גאות. ההתפליגויות הנורמליות נבדלות אחת מהשנייה באמצעות הממוצע וסטיית התקן (-הפרמטרים שמאפיינים את ההתפלגות). התפליגות נורמלית סטנדרטית היא:

התפליגות נורמלית בעלת תוחלת 0 ושונות 1:

$$X \sim \mathcal{N}(0,1)$$

עבור תוחלת ושונות μ, σ^2 , פונקציית הצפיפות של משתנה נורמלי הינה:

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

ניתן להשתמש במומנטים כדי למצוא קשרים בין התפליגויות. למשל עבור שני משתנים המתפלגים נורמלית:

$$X \sim \mathcal{N}(\mu_x, \sigma_x^2), Y \sim \mathcal{N}(\mu_y, \sigma_y^2)$$

המונומנטים מתקיימים:

$$M_X(t) \cdot M_Y(t) = e^{\mu_x t + \frac{1}{2}\sigma_x^2 t^2} \cdot e^{\mu_y t + \frac{1}{2}\sigma_y^2 t^2} = e^{(\mu_x + \mu_y)t + \frac{1}{2}(\sigma_x^2 + \sigma_y^2)t^2} = M_{X+Y}(t)$$

ולכן ניתן לחשב את ההתפלגות של $X + Y$:

$$X + Y \sim \mathcal{N}(\mu_x + \mu_y, \sigma_x^2 + \sigma_y^2)$$

אי שיוניים

מרקוב

בහינת $0 \leq X$, התוחלת $\mathbb{E}[X]$, עברו פרמטר $a > 0$ מתקיים:

$$p(X \geq a) \leq \frac{\mathbb{E}[X]}{a}$$

צ'בישב

בහינת התוחלת $\mathbb{E}[X]$ והשונות $Var[X]$, עברו פרמטר $a > 0$ מתקיים:

$$p(|X - \mathbb{E}[X]| \geq a) \leq \frac{Var[X]}{a^2}$$

צ'רנוף

בහינת התוחלת $\mathbb{E}[X]$, עברו שני פרמטרים $a, t > 0$ מתקיים:

$$p(X \geq a) \leq \frac{\mathbb{E}[e^{tx}]}{e^{ta}} = e^{-ta} M(t)$$

כאשר $M(t)$ היא פונקציה יוצרת מומנטים של X .

ינט

מעבר לשתנה מקרי X בעל תוחלת, עברו פונקציה קמורה $\mathbb{R} \rightarrow \mathbb{R}$: g : מתקיים:

$$g(E[x]) \leq \mathbb{E}[g(x)]$$

התפלגות דו ממדית

$$F_{x,y}(a, b) = P(x \leq a, y \leq b)$$

תכונות:

$$\lim_{a,b \rightarrow \infty} F_{x,y}(a, b) = 1$$

$$\lim_{a \rightarrow -\infty} F_{x,y}(a, b) = \lim_{b \rightarrow -\infty} F_{x,y}(a, b) = 0$$

$$P(c < x < a, d < y < b) = P(x < a, y < b) - P(x < a, y < d) - P(x < c, y < b) + P(x < c, y < d)$$

$$= F_{x,y}(a, b) - F_{x,y}(a, d) - F_{x,y}(c, b) + F_{x,y}(c, d)$$

אם x, y בלתי תלויים אחד מתקיים:

$$\forall a, b \ F_{x,y}(a, b) = F_x(a) \cdot F_y(b)$$

זוג משתנים נקרא דו-ממדי רציף אם קיימות פונקציות צפיפות דו-ממדית $f_{x,y}(s, t)$, כך שמדובר מתקיים:

$$P(x, y \in A) = \int f_{x,y}(s, t) ds dt$$

באופן שקול מתקיימים:

$$f_{x,y}(s,t) = \frac{\partial^2}{\partial s \partial t} F_{x,y}(s,t) = \frac{\partial^2}{\partial t \partial s} F_{x,y}(s,t)$$

התפelogות שלויות:

$$F_x(s) = P(x \leq s) = P(x \leq s, y \leq \infty) = \int_{-\infty}^s \int_{-\infty}^{\infty} f_{x,y}(x,y) dx dy$$

נוסחת ההסתברות השלהמה לציפיות (באופן שקול גם ל- $f_y(t)$):

$$f_x(s) = \frac{d}{ds} F(x_s) = \int_{-\infty}^{\infty} f_{x,y}(s,y) dy$$

כעת ניתן גם לכתוב תנאי שקול למשתנים בלתי תלויים – y, x בלתי תלויים אם ו רק אם:

$$\forall x, y \quad f_{x,y}(X,Y) = f_x(X) \cdot f_y(Y)$$

סטטיסטיקה היסקית

אם ידעים את סוג ההתפelogות אבל לא ידעים את מרכיביה, ניתן לאמוד את המרכיבים בעזרת מדגם. המדגם מאפשר לנו להשתמש באופןן עבור מספר מאורעות שניי התפelogות. דוגמא – נניח קצץ למדוד גובה של קבוצה מסוימת – כל התלמידים בבית ספר מסוים. ידוע שגובה מתפלג נורמלית, אבל לא ידועים כאן את התוחלת והשונות. לשם כך ניתן להשתמש באומד – פונקציה שמנוהת עליה את המאורעות ומתווך כרך להסיק את התוחלת והשונות.

בניתו נצא מנוקודת הנחה שידוע כי הערכים במדגם נלקחים כולם מתוך התפelogות X , השיכת המשפחה של התפelogות שתלויות בפרמטר אחד או יותר שאים ידועים. (למשל $X \sim N(\mu, \sigma^2)$ כאשר μ, σ אינם ידועים). בפועל נתנות n דוגמאות בלתי תלויות מתוך התפelogות: X_1, X_2, \dots, X_n , ו老子ים לאמוד את הפרמטרים הללו ידועים (כפונקציה של הערכים שדגמנו).

אומד בלתי מוטה: אומד מוגדר להיות בלתי מוטה אם התוחלת של האומד שווה לפרמטר אותו מנוטם לאומד, כלומר, אם $\hat{\theta} = \hat{\theta}(\hat{\theta})$, אז האומד הוא חסר הטיה. במילים אחרות – אומד יהיה חסר הטיה אם התוחלת של המשתנה המקרי המחשב לפניו שווה לא- θ עבור כל θ .

דוגמאות לאמדים בלתי מוטים:

אומד בלתי מוטה לתוחלת – ממוצע חשבוני:

$$\begin{aligned} \hat{\theta} &= \frac{1}{n} \sum_{i=1}^n x_i \\ \mathbb{E}(\hat{\theta}) &= \mathbb{E}\left[\frac{1}{n} \sum_{i=1}^n x_i\right] = \frac{1}{n} \sum_{i=1}^n \mathbb{E}[x_i] = \mathbb{E}[x_i] = \theta \end{aligned}$$

אומד בלתי מוטה לשונות:

$$\mathbb{E}[s^2] = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$$

הוכחה:

$$\mathbb{E}[s^2] = \mathbb{E}\left[\frac{1}{n-1} \cdot \sum_i (x_i - \bar{x})^2\right] = \frac{1}{n-1} \sum_i \mathbb{E}(x_i - \bar{x})^2$$

$$\begin{aligned}
& \frac{1}{n-1} \sum_i \mathbb{E}[(x_i - \mu) - (\bar{x} - \mu)]^2 \\
&= \frac{1}{n-1} \sum_i \mathbb{E}[(x_i - \mu)^2 - 2(x_i - \mu)(\bar{x} - \mu) + (\bar{x} - \mu)^2] \\
&= \frac{1}{n-1} \sum_i \mathbb{E}[(x_i - \mu)^2] - \mathbb{E}[2(x_i - \mu)(\bar{x} - \mu)] + \mathbb{E}[(\bar{x} - \mu)^2] \\
&= \frac{1}{n-1} \sum_i \sigma^2 - 2 \left(\frac{1}{n} \sum_j \mathbb{E}[(x_i - \mu)(x_j - \mu)] + \frac{1}{n^2} \sum_j \sum_k \mathbb{E}[(x_j - \mu)(x_k - \mu)] \right) \\
&\quad \frac{1}{n-1} \sum_i \left[\sigma^2 - \frac{2\sigma^2}{n} + \frac{\sigma^2}{n} \right] \\
&= \frac{1}{n-1} \sum_i \left[\frac{(n-1)\sigma^2}{n} \right] = \frac{n-1}{n(n-1)} \sum_i \sigma^2 = \sigma^2 \blacksquare
\end{aligned}$$

אומד נראות מרבית – (MLE)

בහינת סדרת דגימות מתוך התפלגות עם פרמטר לא ידוע, נגדיר את פונקציית הנראות שלחן מכפלת ההסתברויות של כל הדגימות, או "הנראות של המדגם":

$$L(x_1, x_2 \dots x_n | p(\theta)) = \prod_i P_\theta(x_i)$$

זהוי פונקציה הנקראת של הדגימות והן של הפרמטר.

אם ההתפלגות רציפה מגדרים במקום זאת את פונקציית הנראות להיות מכפלת הצפיפות:

$$L(x_1, x_2 \dots x_n | p(\theta)) = \prod_i f_\theta(x_i)$$

אומדן הנראות המקסימלית הוא פשוט הערך של הפרמטר שמקסם את פונקציית הנראות. כלומר, $\hat{\theta}$ הוא אומדן נראות מקסימלי עבור θ אם $(\hat{\theta}) = \arg \max_{\theta} L(x_1, x_2 \dots x_n | p(\theta))$.

מכיוון שה-log הינה מונוטונית, למקסם את L שקול למקסם את $\log(L)$, וזה לרוב יותר קל, מכיוון שהמכפלה הופכת לסכום:

$$\log L(x_1, x_2 \dots x_n | p(\theta)) = \sum_{i=1}^n \log f_\theta(x_i)$$

נראה מספר דוגמאות לחישוב ה-MLE:

a. מציאת הפרמטר λ בהתפלגות פואסונית:

$$X \sim poi(\lambda)$$

שלב א' – נגדיר את אומדן הנראות – $L = (x_1, x_2 \dots x_n | p_\lambda) = \prod_i P_\lambda(x_i)$. בהתפלגות פואסונית מקיימת:

$$P(X = k) = \frac{e^{-\lambda} \lambda^k}{k!}$$

$$\prod_i P_\lambda(x_i) = \prod_i \frac{e^{-\lambda} \lambda^{x_i}}{x_i!}$$

מסובך למצאו לזה מקסימום, לכן נוציא לוג:

$$\ln\left(\prod_i \frac{e^{-\lambda} \lambda^{x_i}}{x_i!}\right) = \sum_i \ln\left(\frac{e^{-\lambda} \lambda^{x_i}}{x_i!}\right)$$

$$= \sum_i \ln(e^{-\lambda} \lambda^{x_i}) - \ln(x_i!) = \sum_i \ln(e^{-\lambda}) + \ln(\lambda^{x_i}) - \ln(x_i!) \\ = \sum_i x_i \ln(\lambda) - \lambda - \ln(x_i!)$$

cutet nazora:

$$\frac{\partial L}{\partial \lambda} = \sum_i \frac{x_i}{\lambda} - 1 = \sum_i \frac{x_i}{\lambda} - \sum_i 1 = \sum_i \frac{x_i}{\lambda} - n$$

וכשנשווה ל-0 נקבל:

$$\sum_i \frac{x_i}{\lambda} = n$$

ובודד את הפרמטר אותו מנסים לאמוד:

$$\lambda = \frac{\sum x_i}{n}$$

וקיבילנו אומד עבור הפרמטר λ , כאשר נתון סט התוצאות, פוטו נציג אותן, ונמצא מפורשות את הערך של האומד.

זה בעצם תהליכי מציאת MLE .Cutet לבדוק האם האומד הוא מותה או לא, כאשר השתמש בעובדה שעבור התפלגות פואסונית $\lambda = \mathbb{E}(x)$:

$$\mathbb{E}(\lambda) = \mathbb{E}\left(\frac{\sum_i x_i}{n}\right) = \frac{1}{n} \sum_i \mathbb{E}[x_i] = \frac{n\lambda}{n} = \lambda$$

קיבלנו שתווחת האומד שווה לפרמטר, ולכן הוא בלתי מותה.

ב. התפלגות נורמלית:

$$X \sim (\mu, \sigma^2)$$

פה יש שני פרמטרים לאמוד – התוחלת והשונות. ראשית נגידיר את הנראות:

$$f(X) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$$

לכן הנראות תהיה (נשים לב שהמכפלה תעבור לסכום במערך של האקספוננט):

$$\prod_i f(x) = \prod_i \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x_i-\mu)^2} = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n e^{-\frac{1}{2\sigma^2}\sum_i(x_i-\mu)^2}$$

נציא לוג:

$$\ln(L) = \ln\left(\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n\right) + \ln\left(e^{-\frac{1}{2\sigma^2}\sum_i(x_i-\mu)^2}\right) \\ = n \ln\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \ln\left(e^{\frac{1}{2\sigma^2}\sum_i(x_i-\mu)^2}\right)$$

נשים לב שבביטוי הראשון מה שיש בתוך ה- \ln זה בעצם $(2\pi)^{-\frac{1}{2}} + (\sigma^2)^{-\frac{1}{2}}$, ואז המעריך יכול לרדת מוחץ ל- $-\infty$:

$$= -\frac{n}{2} \ln(2\pi) - \frac{n}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} \sum (x_i - \mu)^2$$

כעת בשבייל לאמוד את התוחלת יש לגזר לפי μ , וכך לאמוד את השונות יש לגזר לפי σ^2 :

$$\frac{\partial L}{\partial \mu} = -\frac{(-2)}{2\sigma^2} \sum_i (x_i - \mu) = \frac{1}{\sigma^2} \sum_i (x_i - \mu)$$

וכשנשווה ל-0 נקבל:

$$\hat{\mu} = \frac{\sum_i x_i}{n}$$

ניתן להבחין כי עבור התוחלת האומד הוא בעצם הממוצע של המדגמים. אפשר לבצע תהליכי דומה על השונות, ומתקיים הביטוי:

$$\hat{\sigma}^2 = \frac{\sum_i (x_i - \hat{\mu})^2}{n}$$

1. References

Intro:

<https://www.analytics.org.il/2019/12/ai-vs-deep-learning-vs-machine-learning/>

2. Machine Learning

2.1 Supervised Learning Algorithms

2.1.1 Support Vector Machines (SVM)

Support Vector Machine (SVM) הינו מודל למידה מונחית המשמש ניתוח נתונים לצורכי סיווג, חיזוי ורגרסיה. המודל מקבל אוסף של דוגמאות מתוויות במרחב d -ממדי, ומנסה למצאו משורר המפריד בצורה טובה כמה שנית בין דוגמאות האימון השיכנות לדוגמאות השונות.

המשמעות הנוצר בא��ון מודל SVM הינו לנדרי, כאשר חלוקת הדוגמאות במרחב הוקטור נועשית באופן כזה שיוציא מרוחק גדול ככל האפשר בין המשורר המפריד לבין הקווים המוקומות היכי קרוב אליו. מרוחק זה מכונה שולים (margin), אשר בצד אחד של השולים נמצאות דוגמאות עם label אחד, ובצד השני נמצאות הדוגמאות עם ה-label השני. את המשורר המפריד ניתן לייצג באמצעות אוסף הקווים \vec{w} והמתקיימות $0 = \vec{x} \cdot \vec{w} + b$, כאשר \vec{w} הוא וקטור נורמלי של המשורר.

נסח את האלגוריתם באופן פורמלי: נתן אוסף של n נקודות (x_i, y_i) , כאשר $y_i \in \{-1, 1\}$. התיוג המתאים לדוגמא i , ו- $x_i \in \mathbb{R}^d$ הוא וקטור המאפיינים המתוארים את דוגמא i . מודל ה-SVM מייצר משורר המפריד את המרחב לשני מרחבים שכלי אחד מהם אמר להכיל עיקרי דוגמאות מסווג תיוג אחד. בנוסף, המודל מייצר שני משוררים מקבילים לו, אחד מכל צד, במרקח דקה וגדול ככל האפשר:

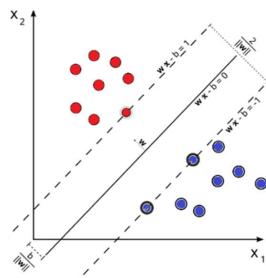
$$\vec{w}^* = \operatorname{argmin}_{\vec{w}} \left(\frac{1}{2} \|\vec{w}\|^2 \right), \text{s.t. } y_i(\vec{w} \cdot \vec{x}_i) \geq 1$$

כלומר, רצים למציאת וקטור המשקלות \vec{w} המשורר שולים $margin = \frac{1}{2} \|\vec{w}\|^2$ המתאימים לכך שהדוגמאות מתוויות נכון ($0 < y_i(\vec{w} \cdot \vec{x}_i) < \frac{1}{2} \|\vec{w}\|^2$) ולא מתקיים ($y_i(\vec{w} \cdot \vec{x}_i) \geq 1$).

ישנו מספר גישות למציאת המפריד, ונפרט על כמה מהן.

Hard-Margin (hard SVM)

במצב הפשטוט ביותר, המשוואה עבר כל אחד מגדדיי של המפריד הינה פונקציה ליארית של המאפיינים וכל הדוגמאות אשר סוגו נכונה. מצב זה מכונה " הפרדה קשיחה " בו האלגוריתם מוצא את המשורר עם השול הרחב ביותר האפשרי, ולא אפשר לדוגמאות להיות בין הווקטורים התומכים. זוהי למשה הפרדה מושלמת, והווקטורים התומכים הם למעשה הנקודות בקצוות השולים, כפי שניתן לראות באירוע:



איור 2.1. סיווג באמצעות אלגוריתם SVM עם מפריד בעל השולים הרחבים ביותר. הקו האמצעי מיצג את המפריד, הקווים המוקווים מייצגים את משורי השולים. דוגמאות האימון המטלכחות עם משורי השולים נקראות וקטורים תומכים (support vectors), ומכאן נוצר שם האלגוריתם.

את המשוררים בקצוות השולים ניתן לייצג באמצעות $-\vec{b} = \vec{w} \cdot \vec{x}_i - 1$ or $\vec{b} = \vec{w} \cdot \vec{x}_j + 1$. גאומטרית, המרחק בין שני המשוררים הוא $\frac{2}{\|\vec{w}\|}$, וכן על מנת למקסם את המרחק הזה, יש מהיבא למינימום את $\|\vec{w}\|$. על מנת שדוגמאות האימון לא יכללו בשולים המפרדים, יש להוציא אילוץ לכל דוגמא i , באופן הבא:

$$y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1$$

AILUZ זה מחייב שכל דוגמא תימצא בצד הנכון של המפריד. לכן, במקרה זה יש לקיים את הדרישה הבאה:

$$\begin{aligned} \min_{w,b} & \|w^2\| \\ \text{s.t. } & y_i (\vec{w} \cdot \vec{x}_i - b) \geq 1 \quad \forall i = 1 \dots n \end{aligned}$$

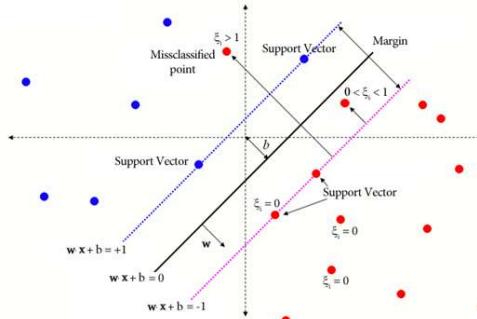
Soft-Margin (soft SVM)

הפרדה מושלמת באמצעות מישור לינארי לעיטים קרובות איננה אפשרית, ולכן ניתן להרחב את המודל כך שיאפשר לנקודות מסוימות לא להיות ב"צד" המתאים להן. ה"פרדה רכה", מאפשרת לטפל בעיות שבחן אין הפרדה לפרטית בין הקבוצות, כמו למשל שיש נקודות חירות. משועת ההרבהה היא שכל וקטור מרופט אחד מהאלומות, אך עם זאת, נרצה להציגו למצב בו האלומות מופרדים "כמה שפחות". הפרדה רכה יוצרת מצב בו יש trade-off בין רוחב השולץ בין השגיאות ומיצאת המשקלים האופטימליים של המסוווג. בגרסת זו יש לרשום באופן מעט שונה את בעיית האופטימיזציה, כאשר מתווסף משתנה המתווסף לנקודות שאינן נמצאות בסיווג המתאים להן לפני המפריד:

$$\begin{aligned} \min & \|w^2\| + C \sum_{i=1}^n \xi_i \\ \text{s.t. } & y_i (\vec{w} \cdot \vec{x}_i - b) \geq 1 - \xi_i \quad \forall i = 1 \dots n, \xi_i \geq 0 \end{aligned}$$

לשם קבלת אינטואיציה, נשים לב לתפקיד המשתנים:

אם $\xi_i = 0$, מתקיים התנאי שנדרש בהפרדה קשיחה, כלומר הנקודה x_i גם נמצאת בצד הנכון של המפריד וגם מתקיימת הדרישה לשמירה על השולדים. אם $0 < \xi_i < 1$ אז הנקודה x_i נמצאת בצד הנכון של המפריד המסווג, אבל המסווג קרוב אליה, כך שהנקודה נמצאת בתוך השולדים. C קבוע שabhängig על "ונישת" של דוגמאות שאין בצד הנכון של המפריד. ערך C גבוה פירושו העדפת הסיווג הנכון על שולדים רבים, ואילו C נמוך מעדיף הכללה (שולדים רבים), גם בבחירה שדוגמאות האימון הספציפיות אין מסוגות נכון.



איור 2.2 סיווג באמצעות אלגוריתם SVM עם הפרדה רכה. המשתנה ξ שווה לאפס אם הנקודה ממוקמת בצד הנכון של המפריד. גודול מאוף כאשר הנקודות נמצאות בצד הלא נכון של המפריד.

Non-linear Separation

מסוגים לינאריים מוגבלים ביכולת ההכללה שליהם啻ם בغالל הפשטות שלהם. לכן, כאשר לא ניתן להפריד אוסף דוגמאות באמצעות מפריד לינארי, משתמשים ב"פרדה א-LINEARIT". גישה זו אפשרית לשימוש ב-SVM לסיווג לא לינארי, על ידי טרנספורמציה לא לינארית, כמו למשל "תעלול הגערין" (Kernel Trick). גישה זו מבצעים מיפוי לדאטה למרחיב אחר, בו ניתן למצוא עוברו הפרדה לינארית, ומילא יי'יה אפשר לשימוש באלגוריתם SVM. כך למשל, קיימת אפשרות ליצור מאפיינים חדשים על ידי העלאת ערכי המאפיינים הקיימים בחזקה מסוימת, הכפלתם בפונקציות טריגונומטריות וכו'.

באופן פורמלי, נחפש פונקציית מיפוי להעתקת מרחב $F \rightarrow \mathcal{H}$: קר שמרחב F ניתן יהיה להפריד את הנתונים $\{(x_i, y_i)\}_{i=1}^N$ באמצעות מסויים לנארו. לשם כך, משתמשים בטריק קרNEL שמקבל כללי וקטוריים מרחב המקורי ומחדר את המכפלה הפנימית (dot product) של הווקטורים למרחב החדש (נקרא גם מרחב התכונות – feature space):

$$K(\vec{x}_i, \vec{x}_j) = \psi(\vec{x}_i)^T \psi(\vec{x}_j)$$

דוגמאות של פונקציות קרNEL נפוצות:

קרNEL לנארו:

$$K(\vec{x}_i, \vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$$

זה הפונקציה המכונה פונקציה פנימית, המוגדרת על ידי מכפלה פנימית של הווקטורים. במקרה זה מרחב התכונות ומרחב הקלט זהים ונחזר לפתרון בעדרת SVM לנארו:

קרNEL פולינומי:

$$K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \cdot \vec{x}_j + c)^d$$

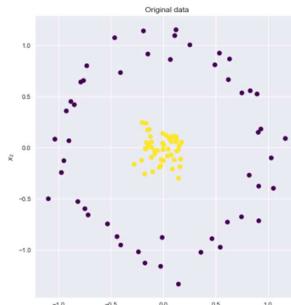
העתקה מהמרחב המקורי למרחב שמהווה פולינום ממעלה 2 $d \geq 2$ הוא פרמטר חופשי המשפייע על היחס בין סדר גובה לעומת סדר נמוך בפולינום. כאשר $c = 0$, הקרNEL נקרא הומוגני.

קרNEL גאוסיאני:

$$K(\vec{x}_i, \vec{x}_j) = \exp(-\gamma(\vec{x}_i - \vec{x}_j)^2), \gamma > 0$$

פרמטר γ מלא תפקיד חשוב, ויש לבחור אותו בהתאם לבעה העומדת בפנינו. אם הערך שלו קטן מאוד, האקספוננט ינהג כמעט לגמרי למרחב אחר גובה יותר ותחל לאלץ מכוחו הלא לנארו. מצד שני, אם געריך אותו יתר על המידה, הפונקציה לא תהיה סדרירה ובול החולטה יהיה רגש מאוד לבנתוני האימוי.

המהות של טרייק קרNEL היא斬立為了 ביצוע את ההעתקה גם מבלי לדעת מה הפונקציה ψ , אלא הידועה של K מספקיה. לצורך קבלת אינטואיציה והמחשה נביא דוגמא. נתן מערך הנתונים הבא:



ניתן לראות שלא ניתן להפריד בין הקבוצות הצביעות בסגולות על ידי מישור הפרדה לנארו. אך נחשש מרחב אחר, מאותו ממד או בעל ממד גבוה יותר, בו ניתן יהיה להפריד בין קבוצות אלה באופן לנארו. לצורך כך נבצע את הפעולות הבאות:

- ניפה את התכונות המקוריות למרחב הגובה 'וותר (מייפוי תכונות).
- מבצע SVM לנארו למרחב החדש.
- נמצא את קבועות המשקלות התואמות את מישור גבול החולטה.
- ניפה את מישור המפריד בחזרה למרחב הדו-ממדי המקורי כדי לקבל גבול החולטה לא לנארו.

שם הרבה מרחבים מממדים גבוהים יותר בהם קוווטות אלה ניתן להפרדה לנארו. נציג דוגמא אחת:

$x_1, x_2 \rightarrow z_1, z_2, z_3$

$$z_1 = \sqrt{2}x_1x_2 \quad z_2 = x_1^2 \quad z_3 = x_2^2$$

למעשה נעדרנו בטריק קרנל. כאמור, בהינתן שקיימת פונקציה $\varphi: \mathbb{R}^n \rightarrow \mathbb{R}^m$ את הוקטורים ממוחב \mathbb{R}^n למרחב תכונות מסוים \mathbb{R}^m , אז המכפלה הפנימית של $x_1 - x_2$ במרחב זהה היא $\varphi(x_1)^T \varphi(x_2)$. קרנל היא פונקציה K השיכת למכפלה פנימית זו, כלומר $(\varphi(x_2)^T \varphi(x_1)) = K(x_2, x_1)$. אם נוכל למצוא פונקציית קרナル המקבילה למפת התכונות שלעיל, יוכל להשתמש בפונקציה ייחד עם SVM ליניארי וכך לבצע את החישובים בעילוות.

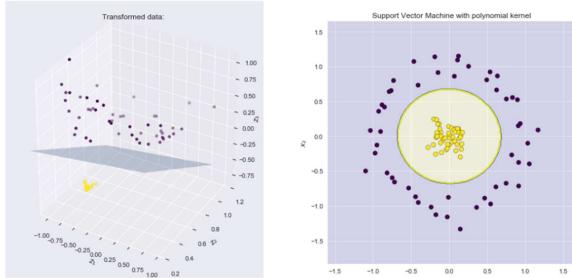
מתברר שמרחיב התכונות שלעיל תואם את השימוש בקרナル פולינומי ידוע: $(x^T x')^d = K(x, x')$. נבחר $d=2$, ונקבל:

$$K\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix}\right) = (x_1 x'_1 + x_2 x'_2)^2 =$$

$$2x_1 x'_1 x_2 x'_2 + (x_1 x'_1)^2 + (x_2 x'_2)^2 = (\sqrt{2}x_1 x_2 x'_1 x'_2) \begin{pmatrix} \sqrt{2}x'_1 x'_2 \\ x_1^2 \\ x_2^2 \end{pmatrix}$$

$$K\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix}\right) = \varphi(\mathbf{x})^T \varphi(\mathbf{x}')$$

$$\varphi\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\right) = \begin{pmatrix} \sqrt{2}x_1 x_2 \\ x_1^2 \\ x_2^2 \end{pmatrix}$$



איור 2.3 שימוש ב-SVM לצורכי הפרדה לאחר ביצוע Kernel trick. המאגל באיר הימני ממופת למישור הפרדה ליניארי במרחב מממד גבוה יותר, כפי שניתן לראות באירוע הימני השמאלי. ניתן לראות שאחרי המיפוי שנעשה באמצעות φ , kernel trick מסויים, הן קודות אכן מופרדות בזירה ליניארית.

2.1.2 Naive Bayes

סיווג בייסיאני הוא מודל המשמש בחוק בייס על מנת לסייע אובייקט $\mathbb{R}^n \in \mathcal{X}$ בעל d מאפיינים לאחת מ- K קטגוריות אפשריות. יחד עם השימוש בחוק בייס, המודל מניח "נאיביות" – בהינתן סיווג של אובייקט מסוים, אין תלות בין המאפיינים השונים שלו.

נניח שיש מודל המקובל וקטור מאפיינים ביאריים {cab ראש, משטעל, חום גבוה}, ומסווג האדם בעל תכונות אלה חוליה בשפעת או לא. באופן כללי ניתן לומר שיש תלות בין שיעול לבן חום גבוה, כלומר העובדה שיש לאדם חום מעלה את ההסתברות שהוא גם משטעל, לмерות זאת, ניתן להגיד באופן "נאיבי" שאם כבר יודע לנו שאדם חוליה בשפעת, אז כבר אין יותר תלות בין היוטו משטעל להיותו בעל חום. באופן פורמלי, אמן סביר להניח שמתפקידים ($\text{משטעל}|p > (\text{חום}| \text{משטעל})$, אך ניתן להניח נאיביות ולקבל: $(\text{שפעת}| \text{משטעל}) = p = (\text{שפעת}, \text{חום}| \text{משטעל})$.

באופן כללי, סיווג בייסיאני נאיבי מניח שהבינהント הסיווג של אובייקט מסוים, המאפיינים שלו בלתי תלויים. הנחה זו כומון לא תמיד מדוייקת, וממילא גם רצוי הסתברויות המבניות ממנה ומשמשים לסייעinem מדוייקים, אך ההנחה

מקלה מאוד על חישוב ההסתברויות של הסיווג הביסיאני, ובמקרים רבים רבים תחת ההנחה זו התקבלו תוצאות סיווג. הסבה להצלחת המודל נועצה בקר שבעביה סיווג העיקר הוא למצוא את הסיווג הסביר ביותר לאובייקט (שפעת או לא-שפעת לנבדק בדוגמה), ולא דואק לקלוט הסתברות מדויקת לכל סיווג. במקרים רבים למרות שההסתברות הנובעת מההנחה הנאיית אינה מדויקת עבור שני סוגי אפסריים, היא בכלל זאת שומרת על סדר ההסתברות שלהם.

נתבונן בוקטור מאפיינים $(x_1, \dots, x_n) \in \mathbb{R}^n = x$, היכל להיות שיר לאחת מ- K קטגוריות $y = (y_1, \dots, y_k)$ התייחסות הפרIORיות y_k ידועה, ובנוסף ידועות המותנות של המאפיינים בהגנת הסיווג – $p(x_i|y_k)$. בעזרת הנתונים האלה רוצים לסואג את x לאחת מהקטגוריות, כלומר למצוא את y שעבורו הביטוי $p(y_k|x)$ הוא מקסימלי. באופן פורמלי ניתן לנוסח זאת כך:

$$y = \arg \max_k p(y_k|x), k = 1 \dots K$$

בשביל למצוא את y_k האופטימלי ניתן להיעזר בחוק ביבס:

$$p(y_k|x) = \frac{p(y_k, x)}{p(x)}$$

המכנה לא תלוי ב- k , ולכן מספיק למצוא את y_k שעבור המונה מקסימלי. לפי כלל השרשרת מתקיים:

$$\begin{aligned} p(y_k, x) &= p(y_k, x_1, \dots, x_n) = p(x_1|y_k, x_2, \dots, x_n) \cdot p(y_k, x_2, \dots, x_n) \\ &= p(x_1|y_k, x_2, \dots, x_n) \cdot p(x_2|y_k, x_3, \dots, x_n) \cdot p(y_k, x_3, \dots, x_n) \\ &= \dots = p(x_1|y_k, x_2, \dots, x_n) \cdot p(x_2|y_k, x_3, \dots, x_n) \cdots p(x_{n-1}|y_k, x_n) \cdot p(x_n|y_k)p(y_k) \end{aligned}$$

icut נשמש בהנחה הנאיית, לפי בהינתן הסיווג y_k , אין תלות בין המאפיינים. לפי הנחה זו נוכל לפשט את הביטוי:

$$\begin{aligned} &= p(x_1|y_k) \cdot p(x_2|y_k) \cdots p(x_n|y_k)p(y_k) \\ &= p(y_k) \prod_{i=1}^n p(x_i|y_k) \end{aligned}$$

בביטוי זה כל האיברים ידועים, ולכן כל שנותר זה רק לחזיב את הנתונים ולקבל את y עבורי ביטוי זה הינו גדול:

$$y = \arg \max_k p(y_k) \prod_{i=1}^n p(x_i|y_k)$$

בדוגמה שהובאה לעיל, המאפיינים קיבלו ערכים בדים, ולכן היה ניתן לחשב את ההסתברות המותנית של כל מאפיין $x_i|y_k$ על ידי ספירת כמה הפעמים שמופיע כל מאפיין באוכליות הנדגת ולחולק בגודל המדגם. עבור ערכים צייפים (כמו למשל מחיר מנתה, גובה של אדם וכדו'), אין אפשרות לחשב כך את ההסתברות המותנית. במקרים כאלה יש להניח התפלגות מסוימת עבור המדגם, וליחסב את הפרמטרים של התפלגות שנות שונות (למשל בעזרת נראות מורכבת – MLE). עבור מדגם המתפלג נורמלית, ההסתברות המותנית היא גאומטרית:

$$p(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} e^{-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}}$$

כאשר σ_y^2 הם הפרמטרים של התפלגות, ואמרם הם משוערים בעזרת MLE או שיטת שערור אחרה. אם ההתפלגות היא לא נורמלית, ניתן להשתמש באלגוריתם Kernel density estimation עבור שערור התפלגות. גישה אחרת להתמודדות עם מאפיינים היכולים לקבל ערכים מצפים היא לבצע דיסקרטיזציה לערכים אותם המאפיינים יכולים לקבל.

במקרה המולטימוד, בו ההתפלגות היא רב ממדית ומציינית תוצאה של סדרה בלתי תלויה, יש לחשב את הנראות באופן המתאים להתפלגות מולטימודית. כדי להבין את החישוב נביא קודם בדוגמה – נניח ורוצים לבנות מודל סיווג בייסיאני המזהה הודעות ספאם. נתנו 12 הודעות, מתוכן 8 אמיתיות ו-4 ספאם. כעת נניח וכל ההודעות מורכבות מארבע של ארבע מילים, בהתפלגות הבא:

Real (R) – {Dear, Friend, Lunch, Money} = {8, 5, 3, 1}.

Spam (S) – {Dear, Friend, Lunch, Money} = {2, 1, 0, 4}.

נחשב את הנראות – ההסתברות של כל מילה בהינתן הסיווג:

$$p(\text{Dear}|R) = \frac{8}{17}, p(\text{Friend}|R) = \frac{5}{17}, p(\text{Lunch}|R) = \frac{3}{17}, p(\text{Money}|R) = \frac{1}{17}$$

$$p(\text{Dear}|S) = \frac{2}{7}, p(\text{Friend}|S) = \frac{1}{7}, p(\text{Lunch}|S) = 0, p(\text{Money}|S) = \frac{4}{7}$$

כעת נבחן מה ההסתברות שהצירוף "Dear friend" הוא מהודעה אמיינית (הצירוף הוא למעשה התפלגות מולטינומית, כיוון שהוא מכיל שתי מילים שאין בין ההסתברויות שלן קשור ישירות):

$$p(\text{Dear friend is R}) = p(R) \cdot p(\text{Dear}|R) \cdot p(\text{Friend}|R) = 0.67 \cdot 0.47 \cdot 0.29 = 0.09$$

$$p(\text{Dear friend is S}) = p(S) \cdot p(\text{Dear}|S) \cdot p(\text{Friend}|S) = 0.33 \cdot 0.29 \cdot 0.14 = 0.01$$

מספרים אלה ניתנים להסיק שהצירוף "Dear friend" אינו ספאם.

באופן כללי, עבור וקטור מאפיינים $x \in \mathbb{R}^n = (x_1, \dots, x_n)$, הנראות מחושבת באופן הבא:

$$p(x|y_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i p(y_{ki})^{x_i}$$

על הציר הלוגריאטרי, בעדרת נוסחה זו ניתן לבנות מסויוג לינארי:

$$p(y_k|x) = \frac{p(y_k, x)}{p(x)} \propto p(y_k) \cdot \prod_i p(y_{ki})^{x_i}$$

$$\rightarrow \log p(y_k|x) \propto \log p(y_k) + \sum_i x_i \cdot \log p(y_{ki}) \equiv b + w^T x$$

היחסורן בשימוש במסויוג בייסיאני נאיבי בעיות מולטינומיות נעוץ בכך שיש הרבה צירופים שלא מופיעים יחד בסוט האימון, ולכן הנראות שלהם תמיד תהייה 0, מה שפוגם באמונות התוצאות.

מקרה דומה להתפלגות מולטינומית הוא מקרה בו המאפיינים הם משתני ברנולי, המתקבלים ערכיהם ביבאים. במקרה זה הנראות הינה:

$$p(x|y_k) = \prod_{i=1}^n p_i^{x_i} (1 - p(y_{ki}))^{1-x_i}$$

עבור דатаה לא מאוזן, ניתן להשתמש באלגוריתם שנקרא (complement naive Bayes (CNB). לפי אלגוריתם זה, במקום ללקחת את $p(y_k)$ נלקחים את המיינומ של הפונקציה ההפוכה:

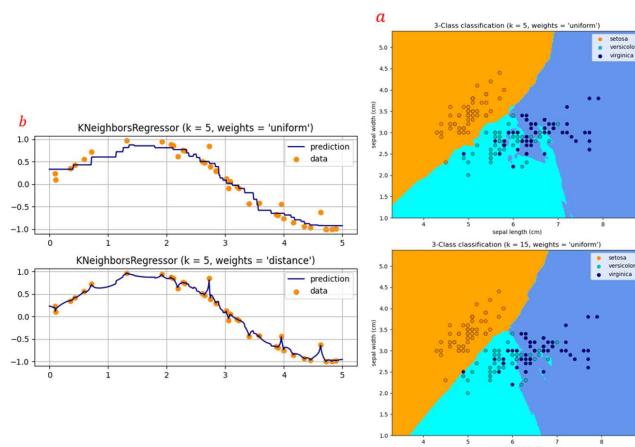
$$\arg \min_k p(y_k) \prod_{i=1}^n \frac{1}{p(x_i|y_k)}$$

שימוש באלגוריתם זה הוכח כיעיל במקרים בהם הדטה אינם מאוזן והביצועים של מסויוגים בייסיאנים אחרים (גאוסיאני או מולטינומי) היה לא מספיק טוב.

2.1.3 K-Nearest Neighbors (K-NN)

אלגוריתם השכן הקרוב הינו אלגוריתם של למידה מונחית, בו נתונים מסטר דומאאות ובונסף יזע ה-label של כל אחת מהן. אלגוריתם זה מתאים להבעיות סיווג (שייך נקודה חדשה למחלקה מסוימת) והן לביעור ורגסיה (נתינת ערך מסוים לנקודה חדשה). האלגוריתם הינו מודול חסר פרמטרים, והוא מבצע סיווג לבתוונים בעזרת הכרעת הרוב. עבור כל נקודה במדגם, המודול בוחן את ה-labels של K הנקודות הקרובות אליו ביותר, ומסווג את הנקודה לפי ה-label שקיבל את מרבית הקולות. מספר הנקודות הקרובות, K, הוא היררכט-פרמטר שנקבע מראש.

אלגוריתם השכן הקרוב הוא אחד המודלים הנפוצים והפשויטים ביותר בלמידת מכונה, כאמור בנוסף לסייע הוא מתאים גם לבעיות גורסיה. המודל פועל בדומה ובשתי המקרים, כאשר ברגסיה יבוצע שקלול של ממוצע בין השכנים הקרובים, ולא הרכעת הרוב, כלומר, התוצאה לא תהיה סיווג label-labels מסוים לפי הערך הפוך בקרבת K השכנים הקרובים, אלא חישוב ממוצע של כל labels השכנים. התוצאה המתבקשת היא ערך רציף, המיצג את הערכים בסביבת התכפייה. ניתן להתחשב במרקח של כל שכן מהטפסית בזורה שווה (uniform), וכן ניתן לחתך משקל שונה לכל שכן בהתאם למרחק שלו מהנקודה אותה רצחים, כך שכל שכן מסוים קרוב יותר לנקודה אותה רצחים מאשר שכן אחר (distance).



איור 2.4(a) סיווג בעדרת אלגוריתם K-NN-מסוגם את המרחב לאזורי בהתאם ל-K השכנים הקרובים בויתר, כך שם TABO גוזה חדשה היא תהיה מסווגת בהתאם לצבע האזור שלא, הנקבע כאמור לפי השכנים הקרובים בויתר. ניתן לראות שיש הבדל בין ערכי K שונים, וכך ש-K יותר גבוהה ככל האזורי יותר חלקים ויש פחות מובלעות. (ב) גורסיה בעדרת אלגוריתם NN-K: קביעת ערך ה- K בהתאם ל-K השכנים הקרובים בויתר. ניתן לחתך משקלים שונים לכל השכנים, או לחתך משקל ביחס לשכן הקרוב אליו רצחים לחשב.

לעתים נאמר על המודל שהוא "עצלן". הסיבה לכך היא שבשלב האימון לא מוחבץ תהליכי שימושותי, מלבד השמה של המשתנים וה-labels-cabiyekim של המחלקה, ככלומר כל נקודה משוכלת למחלקה מסוימת. עקב לכך, כל מדגם האימון (או רובו) נדרש לצורך התחזית, מה שעשוי להפוך את המודל לאטי' כאשר יש הרבה דatasets. למורות זאת, המודל חושב לאחד המודלים הקלאסיים הבולטים, בזכות התיירותו שלו. הוא פשוט וקל לפירוש, עבד היטב עם מספר רב של מחלקות, ומתאים לביעור גורסיה וסיווג. בנוסף הוא חושב אמין במיוחד, כיון שהוא לא מניח הנחות לגבי התפלגות הנתונים (כמו גורסיה ליניארית למשת).

מנגד, יש לו מספר חסרונות. עקב העובדה שהוא דורש את כל נתונים האימון בשבייל התחזית, הוא עשוי להיות איטי כאשר מדובר על דatasets עשיר. מסיבה זו הוא גם אינו יעיל מבחינה זיכרון. מכיוון שהמודל דורש את כל נתונים האימון לצורר המבחן, כisor ההכללה שלו עשיי להיפגם (Generalization). ניקח לדוגמא מורה של ביתה ספר, המנסה לסייע את התלמידים למperfס קבוצות. אם יעשה זאת לפני צבע שער עיניים, חולצה, מכנסיים, נעליים, וכו' – סביר שיתתקה ברכז; אם לעומת זאת הוא ינסה לסייע צבע שער, עיניים, חולצה, מכנסיים, נעליים וכו' – סביר שיתתקל ברכז. במקרה כל תלמיד יחווק מרעהו באופן שואן שני תלמידים שזהוים לחלווטן בכל הפרמטרים, מה שמקשה על חישוב המרחק. בעיה זו מכונה קטלת הממדיות (Curse of dimensionality), וכן מומלץ להיעזר באמצעות להחזרת הממד (Dimensionality reduction).

קיים נוסף הקדים במודל הוא הצורך בבחירת ה-K הנכון, מטלה שעשויה להיות לא קללה לעתים. בכל שימוש של אלגוריתם השכן הקרוב, K הינו היפר-פרמטר שצורך להיקבע מושך. היפר פרמטר זה קובע את מספר המוניות אשר האלגוריתם יתחשב בהן בעת בירית סיווג התכפייה. בירית היפר-פרמטר קטן מדי, לדוגמא K=1, יכולה לגרום למצב בו המודל מותאם יתר על המידה לנ庭וני האימון, מה שਮוביל לדיווק גבוה בנתוני האימון, ודיווק נמוך בנתוני המבחן. מן ההפוך השני, כאשר K גבוהה מדי, למשל 1000, K=100, מוצר המצב ההופך – מודל שמתמחה יותר מדי בדאטא ולא מצליח למצוא הכללה נכונה לסייע. מומלץ לבחור K איזוגי בಗל אופן המבוקש של האלגוריתם – הרכעת

הרב. כאשר בוחרים K זוגי, עלולים להיתקל במצב של שווין אשר עשוי להוביל לתוצאה מוטעית, ולכן כדי להימנע מתיקן כדאי לבחור K אי-זוגי.

כמו אלגוריתמים רבים מבוססי מרחק, אלגוריתם השכנ הקרוב רגש לערכים קיצוניים (Outliers) ושימוש באלגוריתם ללא טיפול בערכים קיצוניים עשוי להשוביל לתוצאות מוטטות. מלבד זאת, חשוב לורמל את הנתונים לפני שימוש במודל, למרות שהאלגוריתם מבוססי מרחק; למשל זה, יתכו מרחוקים בין תוצאות אשר עשויים להשפיע על החלטת המודל, למרות שהם רחוקים אלו הם כמעט מושפעות לצורך הסיג. דוגמא לכך היא משתנה שעשויה לשמש ביחסות מידת שונות (מיילוטקילומטרים). ההחלטה האם להשתמש בקילומטרים או במיילים עולילה להטות את תוצאות המודל, למרות שבפועל לא השתנה דבר.

השיטה הנפוצה ביותר למדידת מרחק בין משתנים רציפים היא מרחק אוקלידי – עברו שתי נקודות שימושו, המרחק ביניהם יוחשב לפי הנוסחה: $\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2} = d$. במידה ומדובר במשתנים בדידים, כגון טקסטים, ניתן להשתמש במטריקות אחרות כגון מרחק המיניג, המודד את מספר השינויים ברכי להפוך מחזרות אחת למחזרות שנייה, ובכך למדוד את הדמיון ביניהן.

לפני שימוש באלגוריתם השכנ הקרוב, יש כרכח לוודא שהמחלקות מואזנות. במידת ומספר דוגמאות האימון באחת המחלקות גבוהה מאשר בשאר המחלקות, האלגוריתם יטה לסייע למחלקה זאת. היסבה לכך היא שיטות מסוימות,เชילוקה או צפיפות להיוות נפוצה הרבה בקרוב K ה샘נים של כל תצפית. הדבר עשוי להביא לתוצאות מוטטות, וכך יש לוודא מראש שקיים שאי-זון בין המחלקות השונות.

2.1.4 Quadratic\Linear Discriminant Analysis (QDA\LDA)

Quadratic Discriminant Analysis הינו מודל נוסף לדיאגנוזות, המניח שההיבטים $x|y_k \sim N(\mu_k, \Sigma_k)$ אובייקט מסוים – מתקבלת התפלגות נורמלית, ככלומר בהינתן $y_k, k \in \{1, \dots, K\}$, מתקיים:

$$x|y_k \sim N(\mu_k, \Sigma_k)$$

ובאופן מפורש, עבור $x \in \mathbb{R}^{n \times d}$ הפלוג המותנה הוא:

$$p(x|y=k; \mu_k, \Sigma_k) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_k|}} \exp\left(-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)\right)$$

בעזרת הנחזה זו, ניתן למצאו מסוג אופטימלי עבור $(x|y_k)p(y_k|x)$. לפיכך ביחס:

$$p(y_k|x) = \frac{p(x|y=k)p(y_k)}{p(x)}$$

המכנה קבוע ביחס ל- y_k, y , ואת $(y|k)$ ניתן לשער בקבילות על פי השכיחות של כל label במדגם – את הביטוי הנוסף במונה – $p(y|x) = p(y|x|y=k) - \text{שאמור מתפלג נורמלית, ניתן לשער בעזרת הנראות מרובית (MLE).}$ נסמן את הפרמטרים של המודל ב: $\{\mu_1, \Sigma_1, \dots, \mu_K, \Sigma_K\} = \theta$, וכן:

$$\theta_{MLE} = \arg \max_{\theta} p(x|y) = \arg \max_{\theta} \log p(x|y; \theta)$$

$$= \arg \max_{\theta} \log \sum_{i=1}^n p(x_i|y_i; \theta)$$

ניתן לפירק את הסכום לפי ה-label של כל דוגמה:

$$= \arg \max_{\theta} \log \sum_{i:y_i=1} p(x_i|y_i=1; \theta) + \log \sum_{i:y_i=2} p(x_i|y_i=2; \theta) + \dots + \log \sum_{i:y_i=K} p(x_i|y_i=K; \theta)$$

כעת בשבייל לחשב פרמטרים עבור כל y_k מספיק להסתכל על הדגימות עבור $k = y$, כלומר:

$$\theta_{kMLE} = \arg \max_{\theta_k} \log \sum_{i:y_i=k} p(x_i|y_i=k; \theta_k)$$

על ידי גזירה והשוויה ל-0 ניתן לחשב את הפרמטרים האופטימליים:

$$\mu_k = \frac{\sum_{i \in y_i=k} x_i}{\sum_i \mathbb{1}_{y_i=k}}$$

$$\Sigma_k = \frac{\sum_{i \in y_i=k} (x_i - \mu_k)(x_i - \mu_k)^T}{\sum_i \mathbb{1}_{y_i=k}}$$

ניתן לשים לב שהתחולلت μ היא למעשה ממוצע הדגימות עבורן $y = k$. בעזרה הפרמטרים המשוערים ניתן לבנות את המסויוג:

$$y = \arg \max_k p(y_k | x; \mu_k, \Sigma_k) = \arg \max_k \log p(x|y=k)p(y)$$

$$= \arg \max_k -\frac{1}{2} \log |\Sigma_k| - \frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) + \log p(y)$$

עבור המקרה בו מטריצת covariance היא אלכסונית, כלומר אין תלות בין משתנים שונים, מתקבל המסווג הביסי אני הගואסיאני (תוצאה זו הגיונית כיון שהמסויוג הביסי אני מניח שהבינהן סיווג של אובייקט מסוים אין יותר תלות בין המשתנים).

עבור המקרה הבינארי, בו $\{0, 1\} \in y$, מתקבל סיווג בצורה של משואה ריבועית:

$$y = 1 \Leftrightarrow -\frac{1}{2} \log |\Sigma_1| - \frac{1}{2} (x - \mu_1)^T \Sigma_1^{-1} (x - \mu_1) + \log p(y=1) > -\frac{1}{2} \log |\Sigma_0| - \frac{1}{2} (x - \mu_0)^T \Sigma_0^{-1} (x - \mu_0) + \log p(y=0)$$

נוסף:

$$a = \frac{1}{2} (\Sigma_1^{-1} - \Sigma_0^{-1})$$

$$b = \Sigma_1^{-1} \mu_1 - \Sigma_0^{-1} \mu_0$$

$$c = \frac{1}{2} (\mu_0^T \Sigma_0^{-1} \mu_0 - \mu_1^T \Sigma_1^{-1} \mu_1) + \log \frac{p(y=1)}{p(y=0)} - \log \frac{\sqrt{\Sigma_1}}{\sqrt{\Sigma_0}}$$

ונקבל:

$$y = 1 \Leftrightarrow x^T a x + b^T x + c > 0$$

זהו משטח הפרדה ריבועי.

-Linear Discriminant Analysis (LDA), בו מוחלטים כי מטריצת covariance זהה לכל ה-labels, כלומר $\Sigma = \Sigma_k$. במקורה זה מתקבל:

$$\log p(x|y=k)p(y) = -\frac{1}{2} (x - \mu_k)^T \Sigma^{-1} (x - \mu_k) + \log p(y)$$

הביטוי $(x - \mu_k)^T \Sigma^{-1} (x - \mu_k)$ נקרא מרחק מהלוביס, והוא מבטא את מידת הקשר בין x לבין μ תוך כדי התחשבות בשונות של כל משתנה. למעשה ניתן להסכך עלי' מסווג המשיר אובייקט ל-label על ידי גזירה והשוויה לא-0 מתקבל השערון:

$$\mu_k = \frac{\sum_{i \in y_i=k} x_i}{\sum_i \mathbb{1}_{y_i=k}}$$

$$\Sigma_k = \frac{1}{n} \sum_i (x_i - \mu_k)(x_i - \mu_k)^T$$

והמסויוג המתקבל הינו:

$$y = \arg \max_k p(y_k | x; \mu_k, \Sigma) p(y) = \arg \max_k -\frac{1}{2} (x - \mu_k)^T \Sigma^{-1} (x - \mu_k) + \log p(y=k)$$

$$= \arg \max_k -x^T \Sigma^{-1} \mu_k + \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log p(y = k)$$

ניתן לסמך:

$$a = \Sigma^{-1} \mu_k$$

$$b = \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log p(y = k)$$

ומתתקבל מסוג לינארי (ומכאן השם של האלגוריתם):

$$y = \arg \max_k a x^T + b$$

משמעות זה מחלק כל שני אזורים בדעתה משור לינארי, כאשר בסך הכל יש קווי הפרדה. עבור המקרה הבינארי מתקיים:

$$a = \Sigma^{-1} (\mu_1 - \mu_0)$$

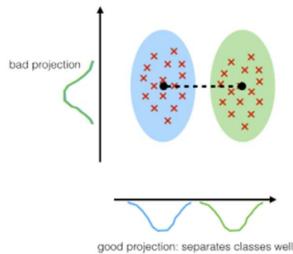
$$b = \frac{1}{2} (\mu_0^T \Sigma^{-1} \mu_0 - \mu_1^T \Sigma^{-1} \mu_1) + \log \frac{p(y = 1)}{p(y = 0)}$$

והסיווג הינו:

$$y = 1 \Leftrightarrow a^T x + b > 0$$

אלגוריתם LDA פשוט יותר מאשר QDA כולל שיטות פרמטרים לשערך, אך יש לו שני חסרונות עיקריים – הוא לא גמיש אלא לינארי, ובנוסף הוא מניח שמט्रיצת ה covariance זהה לכל ה-labels, מה שיכל לגרום לשגיאות בסיווג. כדי להתמודד עם הבעיה השנייה ניתן להשתמש באלאורייטמים המנסים למציאת מטריצת ה covariance-הטובה (Ledoit-Wolf estimator או Oracle Shrinkage Approximating the Oracle Estimator).

באופן גרפי ניתן להסביר על אלגוריתם LDA כיצד כיוון הפרדה בו יש את השונות הגדולה ביותר בין שתי התפלגיות נורמליות, ובנוסף יש בו את הפרדה המקסימלית בין הקבוצות השונות. לאחר מציאת הקוו האופטימלי, ניתן לחשב את התפלגיות השונות של הקבוצות כהתפלגיות נורמליות על הישיר המאנך לקו הפרדה:



איור 2.5 אלגוריתם LDA באופן גרפי: מציאת הכוון של התפלגיות והטלת המידע על הישיר האנכי לכיוון הפרדה.

2.1.5 Decision Trees

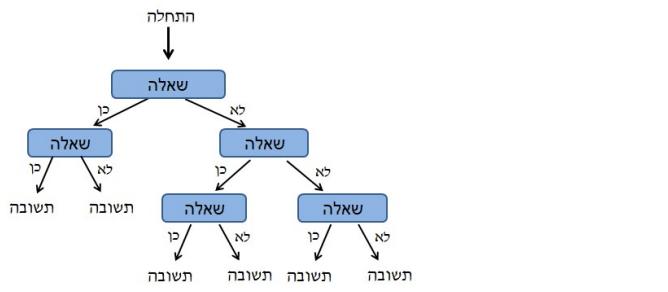
1. קדמה

עוז חלטה הינה לאלגוריתם לומד היכול לשמש הן בעיות סיווג והן בעיות גרגיסיה. באופן כללי, עוז חלטה לוחצים את המשטנה שברצוננו להסביר (משתנה המטרה/החיזוי), ומחלקים את המרחב שלו לקבוצות (segments). לכל קבוצה של סט האימון מחושבים "ממוצע" (Mean) או "שכיח" (Mode), וכאשר מתקבלת ציפוי חדש מש"כים אותה לקבוצה בעלת הממוצע או השכיח הדומה ביזמה. אחר וכליל הסיווג לקבוצות השונות יחולם להציג בקורס עץ, אלגוריתם זה נקרא "עוז חלטה".

הגישהות השונות בעצי החלטה פשוטות ואינטואיטיביות להבנה, אולם הן לא מצלחות להתחזרות במידדי הדיקוק של מודלים אחרים של Supervised Learning. לכן, בפרקם הבאים נציג שיטותensemble, בהן מתבצעת בינוי של כמה עצי-החלטה במקביל, אשר מושלבים בסופו של דבר למודל יחיד. ניתן להראות שימוש במספר גדול של עצים

יכול לשפר דרמטית את מדרדי הדיקט של המודל, עם זאת, ככל שימושים ביותר עצים כך יכולת הפרשנות של המודל נהיית מורכבת יותר ופחות אינטואיטיבית לצופה שאין מעורב במבנה המודל (למשל גורם עסקי בארגון שאנו חווים מועוניים להסביר לו את תוצאות המודל).

ראשית נתבונן במבנה בסיסי של עץ ונגדר עבורי את המושגים הרלוונטיים:



איור 2.6 דיאגרמה של עץ החלטה.

על מנת להבין את מבנה העץ וליצור שפה משותפת נציג את השמות המקובלים בעבודה עם עצים:

- Root (שורש) – נקודת הכניסה לעץ (חלקה העליון ביותר של העץ).
- Node (צומת) – נקודת ההחלטה/פיצול של העץ – השאלות.
- terminal (עלים) – הנקודות של העצים – התשובות. קיימים גם nodes .
- Leaves (ענף) – חלק מסוים העץ המלא (תת-עץ)
- Depth (עומק) – מספר ה- nodes במסלול הארוך ביותר בעץ.

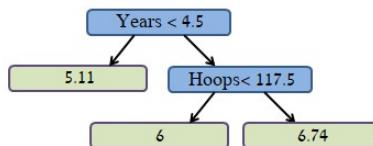
יסודות של עצי החלטה

עצים רגסיביים:

כאמור, עצי החלטה יכולים לשמש הן עבור סיווג והן עבור רגסיה, ונתחיל עם דוגמא פשוטה לבניית עץ עבור בעיית רגסיה – חיזוי שכר (באלפי שקלים) שחקני כדו-שלבי באמצעות עצי החלטה. נרצה לחזות את השכר של שחקן בהתאם על הנתונים הבאים:

- سنים - מספר העונות שאוינו שחקןشيخ בlijgt העל.
- סלים - מספר הסלים שהוא קלע בשעה הקומת.

תחליה נסיר תכיפות ללא ערכים עבור המשתנה המוסבר לנו, הלא הוא "שכר" ובנוסף נבצע על אותו משתנה Log- transform בכך שהוא יהיה ככל הניתן בקרוב להסתגלות נורמלית (עקומת גאות/פערמו).



איור 2.7 דוגמא של עץ החלטה עבור בעיית רגסיה של עץ החלטה.

כאמור, האיור מתאר עץ המונסה לחזות את Log השכר (באלפי שקלים) של שחקן בהינתן מספר עונות הותק שלו וכמויות הסלים שקיים בעונה הקודמת. ניתן לראות שהליך העליון של העץ (Root) מתרחך 2-ענפים. הענף השמאלי מתייחס לשחקנים בעלי ותיק יותר מ-4.5 שנים (< 4.5), הענף הימני מתייחס לשחקנים פחות ותיקים. לעץ יש 2 צמתים (=שאלות/Nodes) ו-3 עליים (=תשומות/Terminal nodes). המספר בכל עלה שבתוחית העץ הוא הממוצע של כל התכיפות ששווגו לעלה זהה. לאחר שבנינו העץ הסטטימאה, כל תכיפות חדשה

שחותוג לעלה מסוים קיבל את הערך הממוצע של התכיפות של בסיסם ומבנה העץ. בהמשך הפרק יסביר כיצד לבנות את העץ וכייד לקובע את כל הפעולות בהתאם לדאטה הקיימ.

בסוף דבר העץ מחלק את השחקנים לשלווש קבוצות:

- שחוקנים ששיחקו 4 עונות או פחות:

$$R_1 = \{X | Years < 4.5, Hoops\} = 5.11$$

- שחוקנים ששיחקו 5 עונות או יותר וקלעו פחות מ- 118 סלים בעונה שחלפה:

$$R_2 = \{X | Years > 4.5, Hoops < 117.5\} = 6$$

- שחוקנים ששיחקו 5 עונות או יותר וקלעו יותר מ- 118 סלים בעונה שחלפה:

$$R_3 = \{X | Years > 4.5, Hoops > 117.5\} = 6.74$$

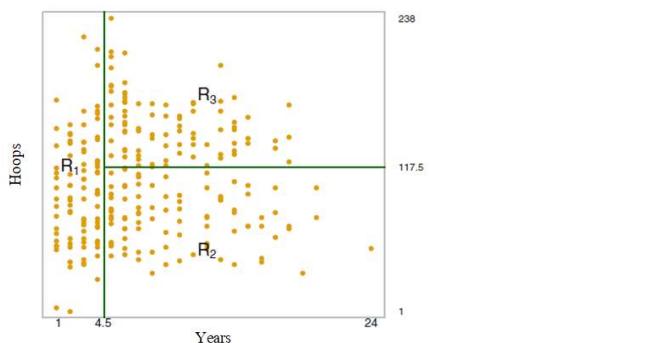
בהתאם לאנלוגיה של העץ, הקבוצות של העץ, R_1, R_2, R_3 מכונות בשם Terminal nodes, או "עלים" של העץ.

אפשר לפרש את עץ הרגסיה שבדוגמה הצורה נוספת: $Years$ הוא הפרמטר המרכזי ביותר בקביעה מה יהיה גובה השכר, שחוקנים עם פחות שנות ניסיון ירוויחו פחות משחקנים מנוסים יותר. עבור שחן חישוב חדש (עד 5 שנים בלבד) הרכטור של כמה סלים יותר האילע בעונה הקודמת מתברר כפחות משיער על השכר. לעומת זאת, לאחרן, ככל עוד לשחקן אין 5 שנים ניסיון, כמה סלים היא ייחסת שולית ביחס לחוסר הניסיון עבור קביעת שכרו. לעומת זאת, בקרוב שחוקנים עם 5 שנים ניסיון או יותר, כמה הסלים שהם קלעו בשנה הקודמת כן משפיעו על השכר, ושחקנים שהקלעו הרבה סלים בשנה הקודמת כנראה ירוויחו יותר כסף מאשר שחוקנים עם אותו ניסיון אך קלעו פחות סלים.

עץ הרגסיה שהובא בדוגמה מפשט קצת "יותר על המידה" את הקשר "האמיתי" בין $Salary$, $years$, $Hoops$ ו- $Years$. והיחס של העץ לא מספיק טוב ביחס למודלים אחרים של רגסיה, כמו למשל רגסיה לינארית שתסביר פרק 3. עם זאת, מודל זה פשוט יותר להבנה ופרשנותו וקליחסו ליצוג באמצעות גרפים.

2. חייזי באמצעות ריבוד (Stratification) של מרחב המשתנים:

עד כה הסביר באופן אינטואיטיבי מהו עץ החלטה וכייד הוא פועל. האתגר המרכזי הוא לבנות את העץ בצורה טובה כך שהיחסיו שלו אכן יהיו תואם למציאות. כדי להבין כיצד עילאה, נציג את הדוגמא הקודמת באופן ניסוי:



איור 2.8 דוגמא של עץ החלטה עבור בעיית רגסיה של עץ החלטה.

באירז זה ניתן לאותו שהנתונים נפרסו על פני מרחב דו ממדי, וחולקו בעזרת קו ההחלטה בהתאם ל- $Years$.Cutת נגידר את תהליך החלוקה והיחסיו בשני שלבים:

1. מחלקים את מרחב הערכים האפשריים של המשתנה אותו רצים לחזות ל- J אזורים נפרדים R_1, \dots, R_J . כתלות בפרמטרים השונים X_1, \dots, X_n .
2. לאחר החלוקה, כל תכיפות שנופלת באחור R_i מקבלת ערך הממוצע של הנקודות מסוימות האימון שמרכיבות את רקבוצת R_i .

באופן תיאורתי, לכל איזור יכולת להיות כל צורה שהיא. אולם לטובת הפשטוט, אנו בוחרים לחלק את המשטנה לא-"[איזורים מרובעים](#)". המטרה היא למצוא את איזורים שambilאים למיניהם את סכום ריבועי ההפחסים בין התוויות של הנתונים בסט האימון לבן הערכים של כל איזור. מודד זה נקרא residual sum of squares (RSS), שמשמעותו:

$$RSS = \sum_{j=1}^J \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

כאשר \hat{y}_{R_j} הוא ממוצע המשטנים של תצפיות האימון באיזור j , y_i הוא המרחק בין כל תצפית לבין הערך הממוצע באיזור זה. כאמור, המטרה של מודד זה היא למצוא את מקצת התצפיות בכל איזור, כך שריבוע המרחק בין הערך הממוצע לבין תצפית יהיה מינימלי. כיוון שבדיקת כל החלוקות השונות האפשרות ייניה ריאלית מבחן חישובית, לרוב משתמשים באלגוריתם חמדן (greedy) אשר עבד "מלמעלה למטה". גישה זו ביחס לעץ החלטה נקראת "פיזול ביניארי ורכוטיבי" (recursive binary splitting), והוא נקראת "מלמעלה למטה" כיון שהוא מתחילה מראש העץ (root), היכן של התצפיות עדין שיינן לקבוצה אחת גדולה. לאחר סימון נקודות ההתחל, האלגוריתם מפצל את מרחב הערכים של המשטנה אותו רוצים לחזות, כאשר כל פיזול מסומן באמצעות שני ענפים חדשים בהמשך העץ.

האלגוריתם כאמור הוא חמדן, וזה מटבטה בכך שככל שבל בתהיל' בניית העץ בוחרים לבצע את הפיזול הטוב ביותר עבור השלב הנוכחי, מבלי להתחשב כמה שלבים קדימה. ניתן לחשוב יוטר על טווח אורך, אך הוא לא יבחר אם הוא לא הפיזול האופטימלי בשלב זה. ניתן להמחיש את דרך הפעולה של אלגוריתם חמדן לעמידה בפקק תנוועה, ומתווך ניסיון לעקוף את הפקק נבחרת הפינה הראשונה שנדרשת פוקה, מבלי להתחשב בפקק שעשו לבוא בהמשך. כמובן שפניה זו לא הכרח תוביל בדרך יוטר יותר מאשריה יותר ארוכת טווח דזוקה היה מוטב להימנע מפניה זו כיון שהיא מובילה לפיקק גדוֹל יוטר בהמשך.

כדי לבצע את אותו "פיזול ביניארי ורכוטיבי", יש לבצע את התהיל' האיטרטיבי הבא:

עבור כל פרמטר אפשרי X_j וקו חילוקה s , נקבל שתי קבוצות:

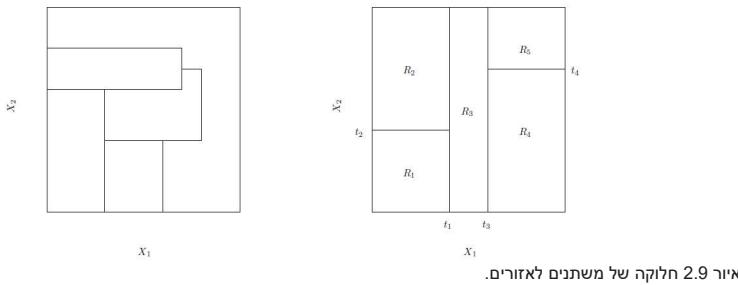
$$\begin{aligned} R_1(j, s) &= \{X | X_j < s\} \\ R_2(j, s) &= \{X | X_j \geq s\} \end{aligned}$$

עבור שתי קבוצות אלה נחפש את הערכים של j ו- s שambilאים למיניהם את המשוואה הבאה:

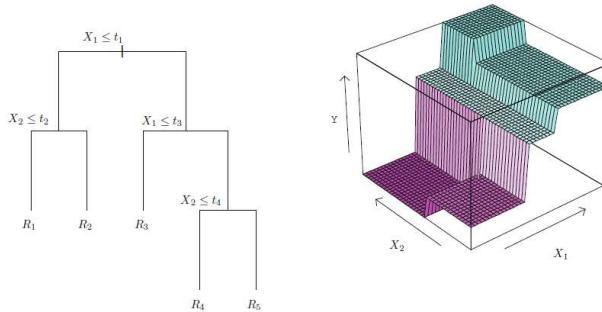
$$\sum_{i: x_i \in R_1(j, s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i: x_i \in R_2(j, s)} (y_i - \hat{y}_{R_2})^2$$

מציאת ערכי s , j שambilאים למיניהם את המשוואה יכולה להתבצע די בקלות כ舍םות הפרמטרים לא גדולה מידי, אך זה תהיל' שלול להיות די סבוך כדיlish כישר הרבה פרמטרים.

למעשה תהיל' זה מייצר ענף אחד שיזוא מה-root, ולן 2 עלים.icut מוצאו את התהיל' שוב, מתור מטרה למצוא את הורמתר הבא שambilאים למיניהם את RSS בכל קבוצה, ובכך לקבל 3 קבוצות. ואוון דומה, ונצה לפאל את אחת מאותן 3 קבוצות שכבר יש לנו כדי למצוות עוד אחת זה RSS. התהיל' הזה נושא איטרטיבית עד שמגיעים "לקיריטוון עזירה". מוסים, כמו למשל מספר פיזולים, מספר מקסימלי של תצפיות בכל איזור וכו'. אורי שהה滂עתה הtolוקה לקבוצות R_j ... R_1 , ניתן להפוך את הקבוצות לעץ, ולהשתמש בו בכדי לחזות נקודות חדשות.



באיור המצורף ניתן לראות שתי חלוקות – החלוקה הימנית הינה חלוקה דו-ממדית של שני משתנים באמצעות פיצול ביןאי רקורסיבי. החלוקה השמאלית היא גם חלוקה דו-ממדית של שני משתנים, אך היא לא יכולה להויצר באמצעות פיצול ביןאי רקורסיבי, כיוון שהיא מורכבת מידי ואינה תואמת את הגישה החמדנית שורזה "חלוקת פשוטה ומהירה".



איור 2.10. תיאור אזרחי חלוקה בעץ ביןאי. מצד שמאל מוצג העץ התואם לחלוקה מהאור הקודם, ומצד ימין ישנה פרנספקטיביה רחבה כיצד חיזויים מתחבאים באמצעות העץ.

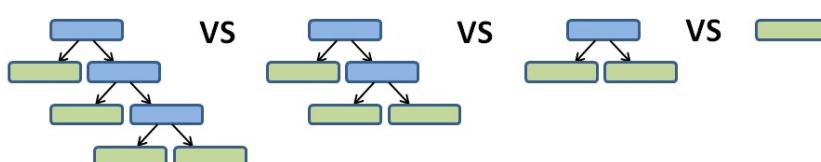
3. גיזום (Pruning):

התהיליך שתואר משקף את התהיליך הקלאסי של יצירת עץ רגסティיה, והוא מסוגל לנפק חיזויים טובים (מבחינת ממד RSS ושונות נמוכה). עם זאת העץ עשוי לבצע התאמת-יתר (Over-fitting) על הנתונים, מה שיוביל לביצועים לא טובים עבור DATA חדש. בעיה זו עלולה לבוא מכך שהעץ שנבנה בתהיליך האימון עשוי להיות "מורכב מדי" ומותאם יתר על המידה לנומני האימון, מה שפוגע ביכולת הכללה שלו. לעומת עץ מרכיב, עץ דיליל יותר בעל פחות פיצולים (כלומר פחות קבוצות R_j , $j=1, \dots, n$) ינתן חיזויים בעלי הטיה (Bias) גדול יותר בהשוואה לאלטרנטיבתה הראשונה, אך כנראה יוביל לשינוי קטן יותר בין סט האימון לבין סט המבחן, ובנוסף מודל זה יהיה קל יותר לפרשנות.

גישה פשוטה בכך להתמודד עם בעיה זו היא לקבוע רף כלשהו, כך שבכל פיצול הרידה ב-RSS תעלה על רף זה. כמובן, פיצול שימושלי לרידה שלויות יחסית ב-RSS לא יתבצע. באופן זה העצים שיתקבלו יהיו קטנים יותר ויש פחות חישש-over-fitting. החיסרון בגישה זו נובע בכך שהיא קיצרת-ארייה. ניתן מצב בו שפיצול בעל תרומה קטנה יחסית אך הוא יכול להוביל לפיצול אחר בעל תרומה גדולה, כה שיביא לרידה גדולה ב-RSS בהמשך. שיטה זו מدلגת על אותו פיצול בעל ערך קטן, מאהר והוא לא עבר את הרף שהוגדר.

גישה אחרת, שמתבסרת והיא טוביה יותר, הולכת בכיוון הפוך. בשלב הראשון יבני עץ גדול מאוד (T_0), ולאחר מכן נגדום ממנו כל מיני פיצולים ב כדי להימנע over-fitting. את מקומם של הנגדומים שhrshtו יתפוז עליה בודד שמקבל ערך ממוצע המתחשב במספר גודלי יותר של תכפיות. כמובן שעוד זה יגרום לרידה ביציעים על פyi סט האימון, אך השאייפה היא שזה 'ישתלם' בבדיקה השגיאה בסט המבחן.

השאלה הגדולה היא כמובן מהי הדרך הטובה ביותר לגוזם את העץ. אינטואיטיבית, המטרה שלנו היא לבחור subtree מסויר העץ המקורי שימושלי לשגיאה הקטנה ביותר. אומדן זה יכול להתבצע על ידי בדיקת השגיאה של העץ החדש ביחס לסט המבחן. כיוון שאין אפשר לבחון את כל תתי העצים האפשריים, נהוג להשתמש בשיטת weakest link pruning (Cost complexity pruning) (זיהוגה גם בשם weakest link pruning). במקרה להתחשב בכל subtree אפשרי, אנו מסתכלים על רצף מסוים של subtrees שהם מעשה חלקים שונים המורכבים את T_0 – העץ המקורי שנבנה בהתאם:

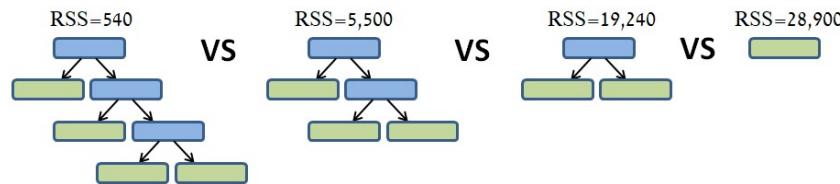


איור 2.11. חיפוש תת עץ אופטימלי ביחס ל- T_0 . מתחילה מ- T_0 (העץ המקורי) וכל פעם מורדים פיצול אחד שמאגים ל-root.

כעת מחשבים עבור כל אחד מרצף העצים שהתקבל את ה-RSS שלו. לפני שנסbir באיזה עץ לבחור, נסכם את השלבים שbowעו עד כה:

- א. בניית עץ ורגסיה גדול מכל הדאטה (ולא רק מסט האימון) בגישה ה- recursive binary splitting (הגיisha החמדנית שתוארה לעיל). העץ ימשיך להבנתה עד קרטירון עצירה מסוים (כמו למשל – הגעה למינימום של צפיפות). עץ זה יסומן על ידי T_0 .
- ב. כל עלה בעץ מקבל את הערך הממוצע של תציפות הפורטר שבאותו עלה, ועbor כל עלה פיצול מחושב ה- RSS.
- ג. סכימת כל ערכי ה-RSS שקיבלו בכל העליים. הסכום שהתקבל הוא ה-RSS של כל העץ T_0 .
- ד. כעת מביצעים את שלבים א-ג' על subtrees הבא ברצף. זהו עץ-משנה שכמעט זהה לעץ המקורי, למעט שני עלים שנגמרו מהעץ המקורי. כך חוזרים על התהליך עבור כל ה- subtrees , עד ל- subtree האחרון שמורכב רק מה-roots של העץ המקורי.

התוצאת של השלב האחרון הוא RSS לכל subtree ברצף העצים. ניתן להבחין כי בכל פעם שענף מסוים הוסר, שהתקבל נהיה גדול יותר לועת subtree-הקוודם. תוצאה זו היגיונית, שהרי כל פיצול בקורסור גורע להקטין את השגיאה באמצעות הקטנת ה-RSS, ואילו גיזום עושה את ההיפר – הוא גורע למנוע over-fitting, גם במחיר של שגיאה גדולה יותר.



איור 2.12 ביצוע גיזום וחישוב RSS לכל עץ (subtree) שהתקבל.

- ה. כעת יש לבחור את אחד מה- subtrees, ובאמת זה נעשה בגלישה pruning. גישה זו משקללת עבור כל עץ את מד ה- RSS שלו יחד עם מספר העליים בעץ. באופן פורמלי:

$$\text{Tree score} = \text{RSS} + \alpha T$$

כאשר T הוא מספר העליים בעץ (Terminal nodes) ו- α פרמטר רגוליזציה הקובע את היחס בין מספר העליים לבין מד ה-RSS. פרמטר זה מחושב באמצעות Cross-validation המבצע trade-off בין הרצון להגדיל את העץ ובכך להקטין את RSS-הו. רצון להימנע עד כמה שניתן מ- over-fitting. המוחזר αT מכונה Penalty, ובאמת הוא גורע "לפצות" על ההפרש במספר העליים השונים. בשלב ד' כבר קיבלנו RSS לכל subtree, וכעת נשאר רק לחשב לכל אחד מהם את ה- Tree score

נשים לב כי:

- כאשר $\alpha = 0$, העץ המקורי (T_0) יהיה בהכרח בעל הנמוך ביותר, כיון שכאשר $0 = \alpha$ אז כל הרכיב αT בנוסחה שווה ל- 0. במקרה זה ה- Tree score יהיה לגמרי למדד ה- RSS:

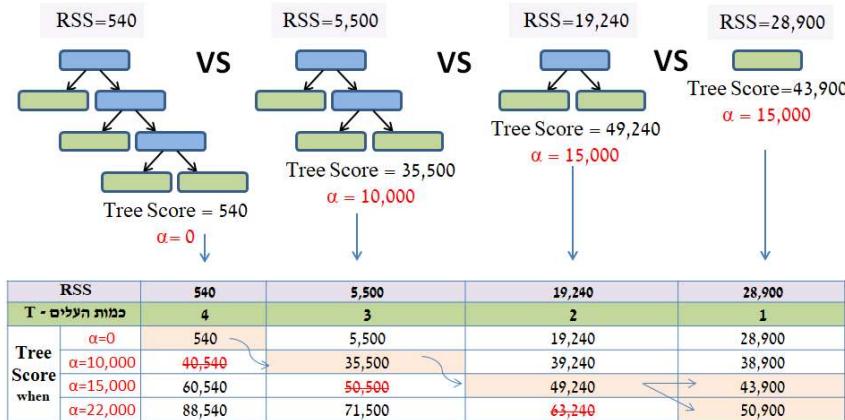
$$\text{Tree score} = \text{RSS} + 0 \cdot T = \text{RSS}$$

ממילא שאשר שלבי החישוב מיותרים, כי אנחנו כבר ידיעים שהעץ T_0 הוא בעל ה-RSS הנמוך ביותר מכל ה- subtrees, וכן בעל ה- Tree Score הנמוך ביותר. לכן, נרצה לקבוע $\alpha > 0$. נתבונן מה קורה ערכיו α שונים:

- ו. עבור $\alpha = 10,000$ הציון של T_0 יהיה 40,540. וכך שווה לנו לאגוז 2 עליים ולקבל עbor אותו α ציון נמוך יותר מ- 40,540 (העץ השני משמאל עם ציון של 35,500). ושוב, אם נשאר ב- $\alpha = 10,000$ ובקביל נגוז 2 עליים נוספים (אנחנו כעת בעץ השלישי משמאל), נקבל ציון של 39,240 – שזה עדין גבוה יותר מ- 35,500. ולכן זה לא טוב לנו.

עבור $\alpha = 15,000$ ניתן לראות שהציגו המתkeletal יהיה נmor יותר מה-subtree הקודם. כעת נשים לב שבמעבר ל-subtree האחרון (ה-root) לא משנה אם α יגדל או ישאר אותו דבר, בכל מקרה הציגו של ה-root יהיה נmor יותר מה-subtree הקודם.

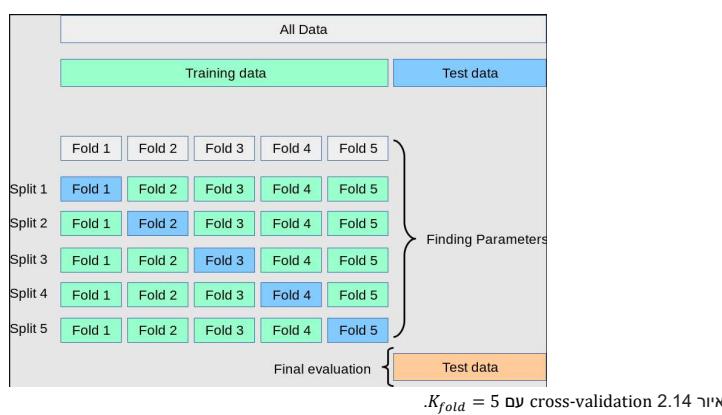
למעשה חזרנו על אותם צעדים עד שאנו יותר מה לגזם. בסופו של תהליך ערכיהם השונים של α יתנו רצף של עצים, מעת מלא ועד לעילו בודד. יוצא מכך שכל ערך של α קיים subtree מתאים $T_0 \subset T$ כך שה-tree score המתקבל יהיה קטן ככל האפשר.



איור 2.13 חישוב ערך α מותאם לכל עם משנה כך שיתקבל קטן ככל האפשר.

העת, בחר בscore-sum subtree-ב- α והנمر ביותר – כלומר העץ השוני משMAIL עם ציון של 35,500. במקביל יש לזכור מהם ערכי α של subtrees השונים שהתקבלו (בסעיף הקודם), כיוון שהם יהיו שימושיים לחישוב ערך α אופטימלי:

- ערכי α -השווים, כיוון שליל α עשוי להתקבל עץ אחר בעל ציון מינימלי. עד לשלב זה העץ בניית הDatasets (בלי חלוקה בין Train ו-Test). אך המטריה היא לבחון מהו ערך α -ה- α שייתן את התוצאות האופטימליות בעדרת העץ הגזום עבור Dataset אחד.
- ט. כדי לבחור את ערך α -האופטימלי ניתן להשתמש ב-Validation-Cross-Validation. לשם כך, יש לקח את הדatasets המלא (ממן בניית העץ הראשון – T_0), ולחلك אותו באופן הבא:



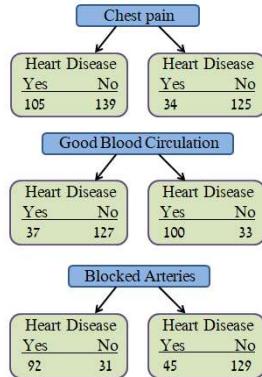
- ב-cross-validation המודל מתאמן תחילה לפי 1 split (איור 2.14 לעיל) על התוצאות שבחלקים הירוקים, ובוחן את החיזויים על סט התוצאות הכלול.
- התהילך זהה נעשה K פעמים, כאשר בכל איטרציה האימון נעשה את התוצאות שבחלקים הירוקים, ובוחינת החיזויים (וחישוב RSS) ועשה על התוצאות שבחלק הכלול.
- . בכל אחד מה-splits ב-cross-validation המודל משתמש רק ב-Training data כדי לבנות עץ מלא (נסמן אותו הפעם ב- T_1 וריצף sequence) חדש של subtrees שմבאים למינימום את-ה-Tree Score (אותו סדר פעולה כמו בסעיפים א'-ד', בעדרת אוטם ערכי α שהתקבלו בסעיף i).
- א. כעת יש לחשב את RSS הנמור ביותר. נניח שבאיטרציה הראשונה העץ שקיבל את RSS הנמור ביותר-Subtree הוא דזוקה העץ שבו $\alpha = \alpha_0$.
- ב. את התהיליך של סעיפים ויא' יש לבצע K פעמים – פעם את העץ עבור כל split, כאשר בכל איטרציה האימון מ被执行 על התוצאות שבחלקים הירוקים, ובוחינת החיזויים (וחישוב RSS) מ被执行 על התוצאות שבחלק הכלול.
- ג. בסופו של התהיליך, כל איטרציה תניב RSS מסוים, וכן ערכי α שנקבעו כבר מראש בסעיף ד'. לאחר K האיטרציות יש לבדוק מיהו העץ עם RSS המינימלי מבין כל האיטרציות, ומהו ערך ה- α של העץ זהה, וזה יהיה הערך הסופי של α .
- ד. לבסוף, יש להזור לעץ המקורי T_0 וה- K subtrees שנבנו מה-set data גם את RSS Train ו-Test ואנו אומרים ש选出 העץ שמתאים לערך ה- α הנבחר. העץ subtree ה- α הנקרא "העץ יפה" שיבחר.

לטיכום:

- אחרי כל החישובים מצאנו שהעץ T_0 בעל RSS גמור ביותר, אך יתרן והוא סבול מ-Overfitting. כדי לפצות על כך, נוסך רכיב שונע להתחשב גם ביתר העצים שהוגנו בעלי RSS גובה יותר, ו- "מעניש" את העץ המקורי (T_0) עקב ריבוי העלים שבו.
- באופן זהה התקבל ציון המאפשר לשוואות בין העצים ולבחור את העץ הטוב ביותר. העץ הזה מהווה מעין איזואן בין הרצון ל-RSS מינימלי לבן שונות נמוכה בין ה-Train ל-Test.
- כדי למצוא את ה- α האופטימלי, שמצד אחדណן עץ בעל RSS אופטימלי ומצד שני נמנע כמה שניתן מ-Overfitting ניתן להיעזר ב-cross-validation.

4. עץ סיווג

עץ סיווג ד' דומה לעץ רגסיה, רק שהמטרה היא שונה – במקומם לקבל תשובה כמותית כמו ברגסיה, עץ סיווג י"ת תווית (label) לתחזית המבוקשת. כאמור לעיל, עץ רגסיה מספק חיזוי לתחזית מסוימת בהתאם לערך המומוצע של תוצאות האימון ששiccות לאותו node terminal. בעץ סיווג לעומת זאת, כל תוצאות תשוויה לקבוצה (class) בעל תווית מסוימת. למשל, נניח ומעוניינים לסתוך מטופל מסוים האם יש לו מחלת לב או לא. אנו יכולים לבנות עץ החלטה על בסיס מאפיינים של חולים שאובחנו בעבר ואנו יודעים להגיד מי מהם באמת חולה לב וכי לא, ועל בסיס העץ הזה להחליט אם מטופל חדש האם הוא דומה במאפיינים שלו למטופלים שאובחנו בעבר חולוי לב או לא. כך שההשובה שהען נותן היא לא "ערך ממוצע" כמו שראינו בעשי רגסיה, אלא פשוט חלהה - "ק חולה לב" או "לא חולה לב". מלבד החיזוי של התווית, עץ סיווג מספק גם יחסים בין הקבוצות השונות בקרבת תוצאות האימון שנופלים באותו אזור. נתבונן בדוגמה שתמחיש את העניין:



איור 2.15 השפעת פרמטרים שונים על הסיכוי לחלות במחלה לב.

באIOR לעיל ניתן לראות שלושה פרמטרים (שבמקרה זה הם סימפטומים של מטופל) בעזרתם מנוטים לסתוג האם למטופל יש מחלת לב או לא. כפי שניתן לראות אף אחד מהמשתנים אין יכול לענות על שאלה זה בפני עצמו, כיוון שבאופן אחד מהעלים אין אחידות בתוצאות. משתנים אלה, אשר אינם יכולים בפניהם לספק סיווג מושלם, נקראים משתנים לא הומוגניים (impure) – לא טהורם. כיוון שרובו המקרים כל המשתנים אינם הומוגניים, יש למצאו דרך כיצד לבחור באחד מהמשתנים להיות המשנה שבראש העץ (Root node). ככלותם, יש ליצור ממד הבחן ומשווה את רמת Entropy-Gini index של כל משתנה. ישים מספר מדרדים, ונתמך בשניים מהם –

א. ממד Gini index:

נסמן את ההסתברות לשירזתוויות מסוימת לקבוצה j ב- $-r_k$, ונגידו:

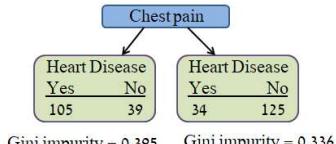
$$Gini = 1 - \sum_{j=1}^J p_j^2$$

באופן אינטואיטיבי, ממד Gini מייצג את הסיכוי לקבלת סיווג שחי עבור בחירה רנדומלית של נקודה מהדעתה בהתאם לפורופרציות של כל class בדата. נציג זאת על אחד הפרמטרים שבודגנו הקודמת – pain. עבור הענף הימני מתקיים:

$$Gini = 1 - \left(\frac{34}{34 + 125} \right)^2 - \left(\frac{125}{34 + 125} \right)^2 = 0.336$$

עבור הענף השמאלי מתקיים:

$$Gini = 1 - \left(\frac{39}{39 + 105} \right)^2 - \left(\frac{105}{39 + 105} \right)^2 = 0.395$$



איור 2.16 חישוב ממד Gini עבור הפרמטר Chest pain.

אחרי שהчисלנו את ה- Gini Impurity לשני העלים, נחשב את ממד Gini הכלול של כל המשתנה Chest Pain. בשביל חישוב זה יש לאזן בין מספר התכפיות שככל עליה, באופן הבא:

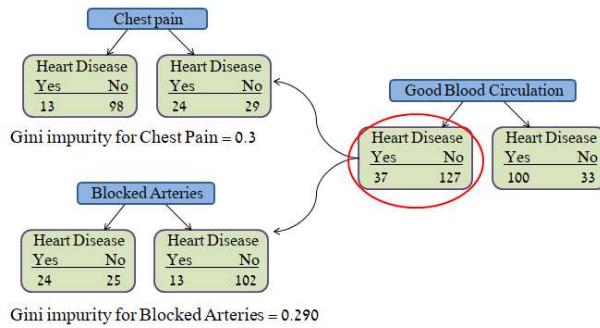
$$\text{Gini impurity for chest pain} = \left(\frac{144}{144 + 159} \right) \times 0.395 + \left(\frac{159}{144 + 159} \right) \times 0.336 = 0.364$$

באופן דומה ניתן לחשב את מzd Gini גם עבור יתר המשתנים ונקבל:

$$\text{Gini impurity for good blood circulation} = 0.36$$

$$\text{Gini impurity for blocked arteries} = 0.381$$

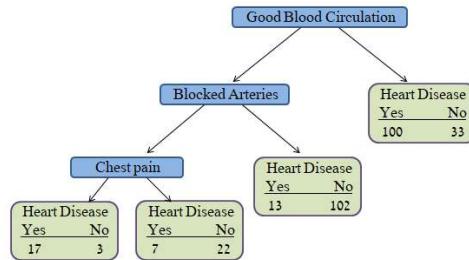
למשתנה Good blood Circulation יש את הציון נמוך, מה שאומר שסוג הci טוב את המטוטלים עם ובל' מחלת לב, ולכן נשתמש בו כמשתנה המסוג הראשון בראש העץ (ה-root). בכך לקבע את הפיצול הבא, יש להתבונן כיצד שאר הפרמטרים מסווגים את התוצאות של ה-root. נניח למשל ומתקיים:



$$\text{Gini impurity for Blocked Arteries} = 0.290$$

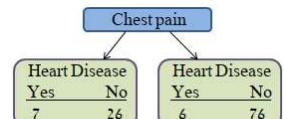
איור 2.17 חישוב מzd Gini עבור שאר הפרמטרים לאחר קביעת ה-root.

כפי שניתן לראות, למשתנה 'Good Blood Circulation' יש ציון נמוך יותר, ולכן זה שנבחר להיות הפיצול הבא. לבסוף נבחן כיצד הפרמטר האחרון מסביר את התוצאות של הפיצול שלפניו, ונקבל את העץ הבא:



איור 2.18 חישוב מzd Gini עבור פיצול לפי הפרמטר Chest pain ביחס לשאר העץ.

נשים לב שניינט לפחות גם את הعلاה 13/102 לפי הפרמטר Chest pain, באופן הבא (המספרים כמובן תלוים בתוצאות האמתיות):



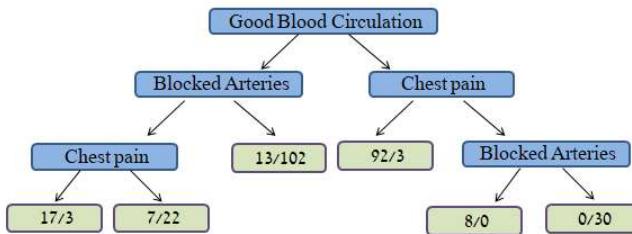
$$\text{Gini impurity for Chest Pain} = 0.29$$

איור 2.19 חישוב מzd Gini עבור פיצול לפי הפרמטר Chest pain ביחס לעלה 13/102.

מדד Gini שהתקבל הינו 0.29, בעוד שבלי הפיצול המדד של העלה היה 0.2, ולכן במקרה זה עדיף להשאיר אותו כי ישיה לפני ניתון הפיצול. כתוב באופן דומה לבנה גם את הענף ימני של העץ:

1. נחשב את מדד Gini.
2. אם לענף הקיים ישין Gini נמוך יותר, אז אין טעם לפצל עוד, והוא הופך ללהיוט terminal node.
3. אם פיצול הענף הקיים מביא לשיפור, בוחרים את המשטנה המפצל בעל ציון Gini הנמוך ביותר.

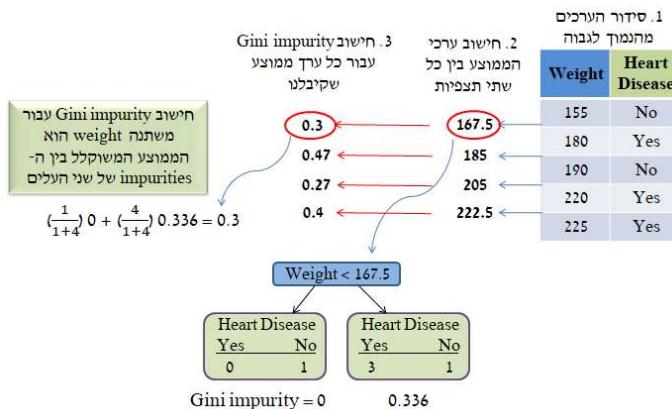
ע"ז טיפולו לאחר סיום התהילה נראה כך:



איור 2.20 חישוב מדד Gini עבור פיצול לפי הפרמטר Chest pain ביחס לעלה 13/102.

מדד Gini שהתקבל הינו 0.29, בעוד שבלי הפיצול המדד של העלה היה 0.2. התהילה שתואר מטהים בינם הפרמטרים מתפצלים באופן יינאר, ככלומר הפיצול של כל פרמטר נקבע על ידי שאלה שעליה יש תשובה של כן או לא. במקרה בהם ישנו משתנים רציפים, הפיצול דורש כמה שלבים מקדים:

1. סידור ערכי הפרמטרים מהערך הנמוך ביותר לערך הגבוה ביותר.
2. חישוב הערך הממוצע בין 2 תוצאות.
3. לחישוב Gini impurity עבור כל ערך ממוצע שהתקבל בשלב הקודם.



איור 2.21 פיצול פרמטרים רציפים לפי מדד Gini.

אנחנו מקבלים את ה-Gini הנמוך ביותר מתי שאנו חנו קובעים threshold של weight<205.

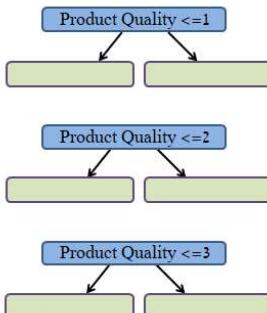
از זהה ה-cutoff וערך ה-Gini שנשתמש בהם כשאנחנו משווים את המשטנה ליתר המשתנים.

עד עכשו דברנו על איר מחלקים משתנה רציף ואיר מחלקים משתנה בנאר (שאלות כן/לא). כתוב הדבר איר מחלק מושתנים קטגוריאליים. למשל: משתנה מודרג שמקבל ערכים מ-1-4. למשל: טיב מוצר מ-1-4. או משתנה שמקיל מספר קטגוריות. למשל: צבע מועדף (מכל ערכיהם: אדום, ירוק, כחול וכו').

משטנה מודרג מודם דומה למשטנה רציף, חוץ מזה שאנו צריכים לחשב בו את הציון עבור כל חלוקה אפשרית.

Product Quality	Satisfied
1	No
1	Yes
3	No
1	Yes
... וכי	... וכי

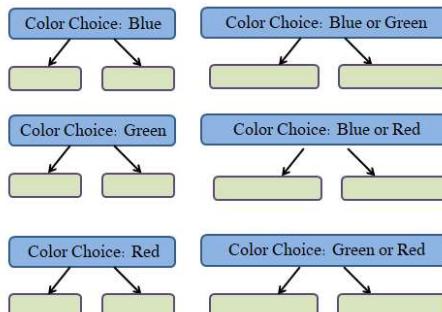
שיעור ל-ב: אנחנו לא צריכים לחשב את ה-
שלילprod. quality score- $=4$ שזה יכול בעצם את כלום



אינטראקטיבי. 2.22 פיצול פרמטרים קטגוריאליים לפי מדד Gini.

कॉम्बिनेशन अपरिहारी: कॉशיש मास्टना कॅटगोरियल उम कमा कॅटगोरियल अपरिहारी, अपर लाचूब अत चिन ह-जून Giin उबोर कॅल कॅटगोरिया, कमो गम उबोर कॅल

Color Choice	Satisfied
Blue	No
Green	Yes
Red	No
Green	Yes
... וכו'	וכו' ...



איור 2.23 פיצול פרמטרים קטגוריאליים לפי מדד Gini.

Entropy & Information Gain מודול ב

ימוד נוסף לפיצול צמות העץ מtbody על האנטורופיה של העלים, בעודו יתן לבחון את ה-*gain information* מהקסימלי מכל פיצול. אנטורופיה באהה למדוד את השגיאה של התפלגות המשטנה הנבחן מול משטנה המטרה. נניח כי ישן n תוצאות אפשריות, כל אחת מהן בעלת הסתברות p_i , אז האנטורופיה מוגדרת באופן הבא:

$$H(x) = - \sum_{i=1}^n p_i \log p_i$$

בזכותם למדד **Gini**, הפיצול האופטימלי נבחר על ידי המשtnה בעל ממד האנטרופיה הנמוך ביותר. אם כל התפצלות בעליה מוסים משוכות לאוטו-class, אז ממד האנטרופיה יהיה 0. מאידך, כאשר בעלה מוסים יש התפלגות שווה בין 2-class-ים של המשtnה המוסבר, ממד האנטרופיה יהיה 1 (זהה הערך המקסימלי שמדד אנטרופיה יכול לקבל).

לעיל פירטו שלב אחריו שלב את חישוב מד Use-Case של חול לבי. בעת ניקח דוגמא אחרת פשיטה יותר הונאות לקבלת מועד לתפקיד מסוים, כאשר מטרת העץ היא להציג "Yes" אם רצים לקבל את המועמד, אחרת "No".



איור 2.23 עץ סיווג עבור קבלת מועד לתקפיך מסוים.

הדגמא מתארת את אחד המאפיינים העיקריים של עצי החלטה – ההחלטה מתתקבלת על ידי הסתכילות היררכית על הפרמטרים השונים, כאשר בכל פעם מתחמדים בפרמטר אחד ואילו כולם בverts אחת. תחילה, נלקח בחשבון המאפיין החשוב ביותר (תואר רלוונטי במקורה שלפנינו), לאחר מכן נלקחים שנות הניסיון, ורק הלאה.

נניח שהסתוכן בקשר לכל העובדים להתקבל ל עבודה הוא 20%, והסתוכן לא להתקבל הוא 80%. במקרה זה האנטרופיה תחשב באופן הבא (הລוגרitem יכל להיות מחושב בכל בסיס, פה נלקח הבסיס הטבעי):

$$H = -0.2 \ln 0.2 - 0.8 \ln 0.8 = 0.5004$$

כעת נניח בנוסך ש-30% מהמועמדים בעלי תואר רלוונטי מקבלים הצעת עבודה ואילו מטור אלה שאינם בעלי תואר רלוונטי רק 10% מקבלים הצעת עבודה. האנטרופיה עbor מועמד בעל תואר הינה:

$$H = -0.3 \ln 0.3 - 0.7 \ln 0.7 = 0.61$$

ועבור מועמד ללא תואר רלוונטי בתחום:

$$H = -0.1 \ln 0.1 - 0.9 \ln 0.9 = 0.32$$

נניח והמועמדים מתפלגים באופן שווה בין בעלי תואר לכלה שאינם בעלי תואר, כלומר ל-50% מהמועמדים יש תואר רלוונטי ול-50% אין, הרי שתווחת האנטרופיה במקורה זה הינה:

$$\mathbb{E}_{H(x)} = 0.5 \times 0.61 + 0.5 \times 0.32 = 0.46$$

לאחר כל החישובים, יוכל לבחון את רמת *the impurity* שמתאפשר מהידיעה האם למועמד מסוים יש תואר רלוונטי או לא. כמובן, כמה הידיעה שלມועמד מסוים יש תואר רלוונטי מפחיתה מהווודאות שלו לקבל את המשרה. אם או הוודאות ננדatta באמצעות מדד Entropy, הר' שהרואה מהמדד הוא:

$$\text{Information gain} = 0.5004 - 0.4680 = 0.0324$$

הן עבור מועמדים בעלי תואר והן עבור לכלה ללא תואר, המשגנה שמקסם את הרוח מהמידע הצפי (ירידה באנטרופיה הצפי) היא מספר שנות הניסיון. כאשר למועדן יש תואר רלוונטי, הסף עbor "שנות ניסיון" הממקסם את הרוח מהמידע הצפי הוא 3 שנים. עבור העף התואם למועדן שאין לו תואר רלוונטי, הסף של שנות ניסיון הממקסם את הרוח מהמידע הצפי הוא 7 שנים. לפיכך, שני הענפים הבאים הם: "ניסיון < 7" ו- "ניסיון ≥ 7". באוטם האוף בונים את יתר העץ.

Misclassification rate

לאחר בניית עץ הסיווג, יש לבדוק את רמת הדיווק שלו על דאטה חדש. בעץ גראסיה זה נעשה בעזרת מדד RSS, ובבעיות סיווג מקובל למדוד את *the Misclassification rate*. מדד זה בא לכמת את היחס בין כמות התוצאות שהמודול סיוג באופן שגוי לכמות הכוללת של התוצאות. במקורה זה פונקציית המחיר תהיה:

$$\mathcal{L}(\hat{y}, y) = I\{\hat{y} \neq y\}$$

פונקציית מחיר זו נקראת loss one-zero, כאשר תחת פונקציה זו חיזיימں נכונים יקבלו ציון 0 ושגיאות יקבלו ציון 1, ללא תלות בגודל השגיאה. פונקציית ה classification rate misclassification rate מוגדרת כ:

$$R(h) = \mathbb{E}[I\{h(x) \neq y\}]$$

סיכום

עż החלטה (Decision Tree) הינו אלגוריתם לסיווג או לחיזוי ערכו של משתנה, כאשר המאפיינים מסוודרים לפי סדר החשיבות. לצורך סיווג, קיימים שני מגדדים אלטרנטיביים לאז' ודוואוט: מדד אנטרופיה (Entropy) ומדד ג'י' (Gini). כאשר ערכו של משתנה מסוים נזהה, אי הוודאות נמדדת באמצעות RSS. חישובו של מאפיין הינו הרוח מהמידע הצפוי שלו (Gain) (Expected Information Gain). הרוח מהמידע הצפוי נמדד על ידי הירידה באז' הוודאות הצפויਆ אשר יתרחש כאשר יתקבל מידע אוזות המאפיין.

במקרה של חלוקת משתנה **קטגוריאלי**, המדע המתתקבל הינו על פי רב אוזות קטגוריה (Label) של המשתנה למשל: צבע מועדף (מכיל ערכים: אדום, ירוק, כחול וכו'). במקרה של משתנה רציף יש לקבוע ערך סף (Threshold) אחד (או יותר) המגדיר שני טווחים (או יותר) עברו ערכי המשתנה. ערכי סף אלו נקבעים באופן שמקסם את ה-information gain הצפוי.

אלגוריתם עż ההחלטה קובע תחילת את צומת השורש (Root Node) האופטימלי של העץ באמצעות קriterיוון "מקסום הרוח מהמידע" שהוגדר לעיל. לאחר מכן הוא מנסה לעשות את הדבר עבור הצמתים העוקבים. הקצוות של הענפים הסופיים של העץ מכונים **צמת עליים** (Leaf Nodes).

במקרים בהם עż ההחלטה משמש לשיווג, צמתי העלים כוללים בתוכם את ההסתבריות של כל אחת מהקטגוריות להיוות הקטגוריה הנכונה. כאשר עż ההחלטה משמש לחיזוי ערך נומרי לעומת זאת, אז צמתי העלים מספקים את ערך התוצאה של העץ. הגיאומטריה של העץ נקבעת באמצעות סט האימון (Training Set), אך הטעיטיסטייקה שweisktת ברמת הדיק של העץ צריכה כמו תמיד בלמידת מכונה לבוא מתוך סט הבדיקה (Test Set) ולא רק מתוך סט האימון.

אחרית דבר:

מדד RSS ומדד misclassification rate שוחצגו בפרק זה הינם רק חלק מהמדדיהם המקוריים. לכל מדד יתרונוטן וחסרונו, והמדד הROLONGNTY יבחר בהתאם לסוג הנתונים והבעיה העסקית. דzon מעת בחוקות וחולשות של עץ ההחלטה:

יתרונות:

- בהשוואה לאלגוריתמים אחרים, עץ ההחלטה דרישים פחות השקעה בתהיליך הכתנת הנתונים (pre-processing).
- עץ ההחלטה לא דורש נרמול של הדטה.
- ערכי חסרים בDATA לא משפיעים על תהליכי בניית העץ.
- עץ ההחלטה תואם לאופן שבו מרבית בני האדם חושבים על בעיה מסוימת והוא פשוט להסביר למי שאינו מומחה.
- אין שום דרישת שהקשר בין המשתנה המוסף והמשתנים המסבירים יהיה לינארית.
- העץ בוחר אוטומטית במשתנים הטוביים ביותר על מנת לבצע את החיזוי.
- עץ ההחלטה רגיש פחות לתחזיות חריגות מאשר רגסיה.

חסרונות:

- שינוי קטן בנתונים יכול לפגוע לשינוי גדוול במבנה עץ ההחלטה ולגרום לחוסר יציבות.
- לעיתים החישוב בעץ ההחלטה יכול להיות מורכב מאוד ביחס לאלגוריתמים אחרים.
- עץ ההחלטה לעתים תכופות דרישים יותר זמן הרצה לאימון המודל, ומשכך מדובר באלגוריתם "יקר" במשמעותו.
- האלגוריתם של עץ ההחלטה אינו מספיק ליישום רגסיה ולণיבוי ערכים רציפים.
- עץ ההחלטה נוטה לעתים תכופות לנוטות ל-Overfitting.

2.2 Unsupervised Learning Algorithms

2.2.1 K-means

אלגוריתם K-means הינו אלגוריתם של למידה לא מונחית, בו מטבחעת תחיזית על נתונים כאשר ה-label (label) אינו נתון. אלגוריתם זה מתאים לביעות של חלוקה לאשכולות (Clustering), ובנוסף יכול לשמש בשלב הצגת ויקי הנתונים (EDA). עבור כל נקודה במדגם, המודל מציין את סכום ריבוע המרחקים (WCSS) מכל מרכז אשכול (Centroid) - (centroid), ולאחר תהליכי של התכנסות – נקבעים האשכולות והסנטרואידים הסופיים. מספר האשכולות הנדרש הוא הירפ-פרמטר שנקבע מראש. כמו כל האלגוריתם השיכים למידה הבלתי-מנוחית, ב-K-means לא מטבחעת אימון, ולמעשה התהזהת מטבחעת על כל הדadata הנתון.

סנטרואיד הוא מונח מתחום היגיומטריה, והוא מתייחס את המוצע האריתמטי של כל הנקודות שמתפרטות על פניהן כולה. באופן אינטואיטיבי ניתן לחשב על סentrואיד נקודות איזון של צורה גיאומטרית כלשהיא, כך שאם נססה להניח צורה, משולש לדוגמא, באופן מאוזן, הסentrואיד הוא הנקודה שבה המשולש יתאזן ולא ייפול לאחדר הצדדים.

בפועל, כדי שהוצאות איזון מתחווים במציאות יותר מרכיבות ממשולש. במקרה זה, הסentrואיד יוויה הנקודה בה סכום המרחקים של כל נקודה באשכול הסentrואיד יהיה מינימלי. לעומת זאת, המודל ממקם את מרכזו של כל אשכול כך שסכום המרחקים של כל הנקודות מהסentrואיד יהיה נמוך ככל האפשר. למשל, זה הגדירה הבסיסית של K-means: אלגוריתם מבוסס סentrואידים המציין את סכום ריבוע המרחק של כל הנקודות באשכול. מגד זה נקרא WCSS, והוא ממד משמעותי ביותר בקשר לאלגוריתמים שבמציעים חלוקה לאשכולות. K-means בפרט, הסיבה להזקה במשוואה היא שגם רצים להציג את ההשפעה של המרחק, מעין "ענן" לציפויו רוחקות מהמרכז.

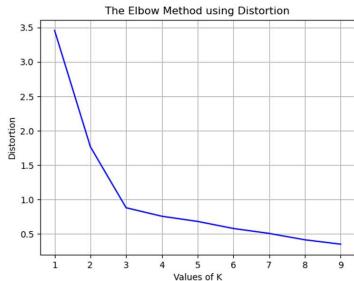
מדד WCSS הוא אחד הדרכים המקבילות ביוטר להעריך את תוצאות החלוקה לאשכולות ב-K-means. היתרון של מדד זה הוא האפשרות לראות באופן מדויק את מידת ההצלחה של המודל, כלומר לפחות ממש' שמכמת את הצלחת המודל. מנגד, WCSS הוא מספר שלא תחום מסוים והוא דרוש פרשנות, כמו שהערך והמשמעות שלו משתנים ממודול למודול. ערך מסוים יכול להיחשב תוצאה טוביה במקרה מסוים, ובמקרה אחר זאת עשויה להיחשב תוצאה רעה מאוד. ניתן להשוות WCSS בין מודלים אך ורק כאשר יש להם אותו מספר אשכולות ואוותם מספר תוצאות. באופן פורמלי, ערך זה מחושב באופן הבא:

$$WCSS = \sum_{j=1}^k \sum_{i=1}^n \|x_i^{(j)} - c_j\|^2$$

כאשר K הוא מספר האשכולות, ו- n הוא מספר הנקודות במדגם.

ישנו trade-off בין השאייה למזער את מדד הרצוי: ככל מספר האשכולות גדול יותר, כך ה-WCSS יקטן. הדבר מתייחס עם ההיגיון – פיזור סentrואידים רבים (כלומר, חלוקה ליוטר אשכולות) על פי הנתונים יוביל לכך שבכrary סכום המרחקים של התוצאות מהסentrואידים יקטן או לא ישנה. כמו שתוצאות מסוימת לסentrואיד הקרוב אליה ביותר, אם התווסף סentrואיד שקרוב לנקודה מסוימת – ה-WCSS יקטן. ואם הסentrואיד רחוק מיל' שאר הנקודות במדגם יותר מהסentrואידים הקיימים – חלוקת התוצאות לאשכולות לא תשנה, וערך ה-WCSS לא ישנה.

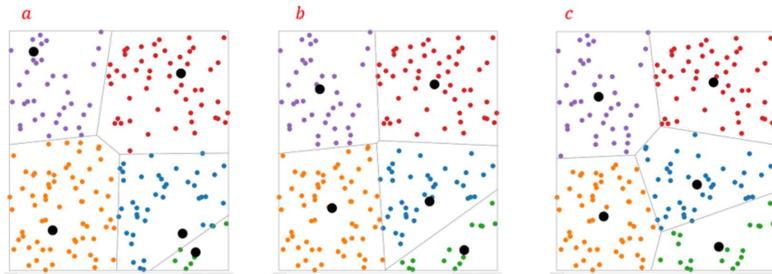
לכן מצד אחד, נרצה לבחור K גדול שמייצג את ה-WCSS; מצד שני, הסיבה שהשתמשנו ב-K-means מילכתייה היא בכך לפשט את הנתונים למספר סביר של אשכולות, כהה שאפשר לנו לעורך אמצעי גונה. שיטת המרפק (Elbow method) היא טכניקה שימושית לפתרון סוגיה זו. הרעיון הוא לבחור את ה-K-הקטן ביותר שמנה השיפור במדד ה-WCSS הוא מטען במידה סבירה. שיטה זו היא היוריסטיבית ואין דרך חד משמעית לקבוע שה-K-הנבחר הוא האופטימלי. בדרך זו ניתן לשכנע מדויק ה-K-שנבחר הוא הנכון, אך החלטה הסופית נתונה לשיקול דעתו של המשתמש.



איור 2.6 Elbow method – שיטה הירטיטית למציאת מספר האשלאות האופטימלי. בדוגמא זו ניתן לראות שבעבר של $K = 2$ - 3 יש ירידת משמעותית בערך של $WCSS$. המעבר מ- 3 - 4 לעומת זאת מוביל לשינוי זניח ב- $WCSS$ (וכך גם בעברים הבאים). לכן ניתן להסיק שבמקרה זה בחרה של $K = 3$ רינה טובה.

כאמור, האלגוריתם מחלק את הנתונים לשתי קבוצות בדרך שמצוירת את סך ריבועי המרחקים של כל צפיפות ממוקד האשכל. באופן פורמלי האלגוריתם מבצע ב-4 שלבים:

- א. **אתחל:** המודל מציב את הסנטרואידים באופן רנדומלי.
- ב. **שייר:** כל צפיפות משיכת לנטראOID הקרוב אליה ביותר.
- ג. **עדכן:** הסנטרואיד מוחזק שכשה- $WCSS$ של המודל יוזערא.
- ד. **חזרה על שלבים b, g עד אשר הסנטרואידים לא זיזים לאחר העדכן,** כולם יש התכנסות.



איור 2.7 אתחול 6 סנטרואידים באופן רנדומלי. (a). שייר כל נקודה לnearestoid הקרוב אליו, ועדכן הסentrואידים לפי מzd ה- $WCSS$. (c) חזרה על b עד להתכנסות.

K -means ידוע בכך שהוא אלגוריתם פשוט ומהיר. לרוב, הבחרה הראשונה בפתרון בעיה של חלוקה לאשלאות תהיה K -means. עם זאת, לאלגוריתם ישנו גם חסרונות. ראשית, בחירת K הנכון עשויה להיות 어렵ה מרבית המקרים. בנוסף, האלגוריתם רגיש מאוד לעריצים קייזרים (Outliers). אופן הפעולה של האלגוריתם מושפע לו ליצור אשלאות רק במצב של ספירות, והדבר אינו אופטימלי ביחס למקרים.

בעיה נוספת להטעור בבחירה המיקום הראשוני של הסentrואידים – כיוון שהבחירה היא רנדומלית, ניתן להיקלע להתקנות במינימום מקומי שהוא אכן המינימום האגובל. כדי להתמודד עם בעיה זה ניתן להשתמש באלגוריתם $K++$. בשלב ראשון האלגוריתם בוחן למקם סentrואיד אחד באופן רנדומלי. לכל צפיפות, האלגוריתם מחשב את המרחק בין הצפיפות לnearestoid הקרוב אליו ביותר. לאחר מכן, צפיפות רנדומלית נבחרת להיות הסentrואיד החדש. הצפיפות נבחרת בהתאם להתפלגות משקללת של המרחקים, כך שיכל שצפיפות יותר רוחקה – קר גובר הסיכוי שהיא תבחר. שני השלבים האחרונים נמשכים עד שנבחרו K סentrואידים. כאשר כל הסentrואידים מוקמו, מבצעים דילוג על שלב האתחול (השלב בו ממקים את הסentrואידים). $K++$ מוביל להתכנסות מהירה יותר, ומוריד את הסיכוי להתקנס לאופטימום מקומי.

2.2.2 Mixture Models

אלגוריתם K -means מחלק n נקודות ל- K קבוצות על פי מרחק של כל נקודה ממרכז מסוים. בדומה ל- K -means גם K -means model הוא אלגוריתם של clustering, אך במקום לסתוכל על כל קבוצה של נקודות כשייכות למרכז מסוים, המודל משיר נקודות להתפלגות שונות. המודל מניח שכל קבוצה היא למעשה דגימות של התפלגות מסוימת, וכל הדעתה הוא ערבות דגימות מספר התפלגות. הקושי בשיטה זה הוא האתחול של כל קבוצה – כיצד

נition לדעת על איזה דוגמאות לנסות ולמצוא התפלגות מסוימת? עקב בעיה זו, לעיתים משתמשים קודם באלגוריתם K-means על מנת לבצע חלוקה ראשונית לקבוצות, ולאחר מכן מנסים למצוא לכל קבוצה של נקודות התפלגות מסוימת.

ראשית נניח שיש k אשכולות, אז נוכל לרשום את ההסתברות לכל אשכול:

$$p(y = i) = \alpha_i, i = 1, \dots, k$$

וכמוכן לפי חוק ההסתברות השלים מתקדים $\sum_i \alpha_i = 1$.

בנוסף נניח שכל אשכול מתפלג נורמלית עם פרמטרים (μ_i, σ_i) , אז נקודה השיכת לאשכול i מקיימת:

$$x|y = i \sim \mathcal{N}(\mu_i, \sigma_i), i = 1 \dots k$$

אם מגיעה נקודה חדשה ורוצים לשער אותה לאחד האשכולות, אז צריך למשהו למצוא את האשכול i שעבורו הביטוי $p(y = i|x) = p$ הוא הכי גדול. לפי חוק ביס' מתקדים:

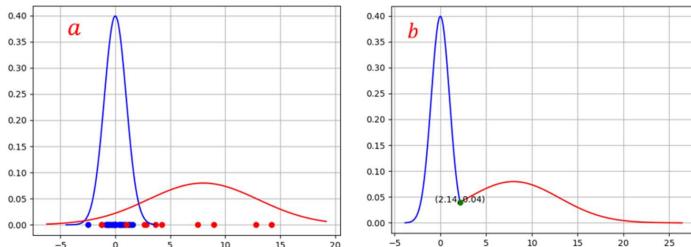
$$p(y = i|x) = \frac{p(y = i) \cdot p(x|y = i)}{p(x)}$$

המכנה למשהו נתון, כיוון שההתפלגות של כל אשכול ידועה ונוטר לחשב את המכנה:

$$f(x) = f(x; \theta) = \sum_i p(y = i)f(x|y = i) = \sum_i \alpha_i \mathcal{N}(x; \mu_i, \sigma_i)$$

ובסע הצל:

$$p(y = i|x) = \frac{\alpha_i \cdot \mathcal{N}(x; \mu_i, \sigma_i)}{\sum_j \alpha_j \mathcal{N}(x; \mu_j, \sigma_j)}$$

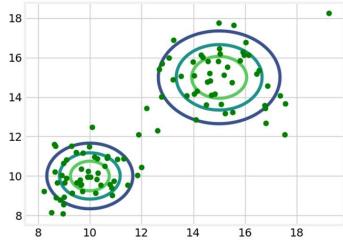


(a) תרורבת של שני גאים באמוד אחד: בשלב ראשון מחלקים את הנקודות לשני אשכולות ונתמנים לכל אשכול התפלגות מסוימת. בקרה זה אשכול אחד (סומן בכחול) הוחזק לההיפות $\mathcal{N}(0, 1)$, ואשכול אחד (סומן באדום) הוחזק לההיפות $\mathcal{N}(8, 5)$.
(b) נקודה חדשה x תסוויג לאשכול הכלול אם $x < 2.14$, כיוון שבתחום זה $\mathcal{N}(0, 1) > \mathcal{N}(8, 5)$. באופן דומה, הנקודה x תסוויג לאשכול האדום אם $x > 2.14$, כיוון שבתחום זה $\mathcal{N}(0, 1) < \mathcal{N}(8, 5)$.

כאמור, כדי לשער נקודה חדשה x לאחד מהאשכולות, יש לבדוק את ערך ההסתברות בנקודה החדש. שבעוריה ההסתברות (x) היא האגדולה ביותר, היא זאת שאיליה תהיה משיכת הנקודה. ההסתברויות יכולות להיות בכל ממד, אך הן יכולות להיות גם בממד יותר גבוה. למשל אם מסתכלים על מישור, ניתן להעתים לכל אשכול התפלגות נורמלית דו-ממדית. במרקחה ה-a ממד', התפלגות נורמלית (Σ, μ) היא בעלות האפיות:

$$f_X(x_1, \dots, x_n) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} e^{-\frac{1}{2}(X-\mu)^T \Sigma^{-1} (X-\mu)}$$

כאשר $|\Sigma|$ הוא הדטרמיננטה של מטריצת ה-covariance.



אור 2.9 תערובת של שני גaussians בדו-ממד: אשלול אחד מתאים לאויאן עם וקטור תוחלת $\mu_1 = [10, 10]$ ומטריצת covariance: $S = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$, והאשלול השני מתאים לאויאן עם וקטור תוחלת $\mu_2 = [15, 15]$ ומטריצת covariance: $S = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$.

כיוון שהאלגוריתם mixture model מספק מטפסה בו כמודל גנרטיבי, ניתן להשתמש כל אשלול גנרטיבי, כלומר מודל שיודע לייצר דוגמאות חדשות. לאחר התאמת הפלגות לכל אשלול, ניתן לדגום מההפלגות השונות ובכך לקבל דוגמאות חדשות.

2.2.3 Expectation–maximization (EM)

אלגוריתם מקסום התוחלת הינו שיטה איטרטיבית למציאת הפרמטרים האופטימליים של הפלגות שונות, במרקם בהם אין נוסחה סגורה למציאת הפרמטרים. נתבונן על מקרה של Mixture of Gaussians, ונניח שיש אשלול מסוים המתפלג נורמלית עם תוחלת ושותות (μ, σ^2) , ומשוכנת אליו n נקודות. כדי לחשב את הפלגות של אשלול זה ניתן להשתמש בולוג הנראות המרבית:

$$L(\theta|x_1, \dots, x_n) = \log \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} = \sum_{i=1}^n \log \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{(x_i-\mu)^2}{2\sigma^2}$$

כדי למצוא את הפרמטרים האופטימליים ניתן לגזר ולהשוו ל-0:

$$\frac{\partial L(\theta)}{\partial \mu} = \sum_{i=1}^n \frac{x_i - \mu}{\sigma^2} \rightarrow \mu_{MLE} = \frac{1}{n} \sum_{i=1}^n x_i$$

$$\frac{\partial L(\theta)}{\partial \sigma^2} = \frac{1}{2\sigma^2} \left(-n + \frac{1}{\sigma^2} \sum_{i=1}^n (x_i - \mu)^2 \right) \rightarrow \sigma_{MLE}^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2$$

כעת נניח ויש k אשלולות וכל אחד מתפלג נורמלית.Cut שט הפרמטרים אותם צריך להעריך הינו:

$$\theta = \{\mu_1, \dots, \mu_k, \sigma_1^2, \dots, \sigma_k^2, \alpha_1, \dots, \alpha_k\}$$

עבור מקרה זה, הלוג של פונקציית הנראות המרבית יהיה:

$$L(\theta|x_1, \dots, x_n) = \log \prod_{i=1}^n \sum_{j=1}^k \alpha_j \mathcal{N}(x_i, \mu_j, \sigma_j^2) = \sum_{i=1}^n \log \left(\sum_{j=1}^k \alpha_j \mathcal{N}(x_i, \mu_j, \sigma_j^2) \right)$$

אם נגזר ונשווה ל-0 נקבל בדומה למקרה פשוטו:

$$\sum_{i=1}^n \frac{1}{\sum_{j=1}^k \alpha_j \mathcal{N}(x_i, \mu_j, \sigma_j^2)} \alpha_j \mathcal{N}(x_i, \mu_j, \sigma_j^2) \frac{(x_i - \mu_j)}{\sigma_j^2} = 0$$

נוסחה זו אינה ניתנת לפתרון אנליטי, ולכן יש הכרח למצוא דרך אחרת כדי לחשב את הפרמטרים האופטימליים של הפלגות הרצויות. נתבונן בחלק מהבטי שקבענו:

$$\frac{1}{\sum_{j=1}^k \alpha_j \mathcal{N}(x_i, \mu_j, \sigma_j^2)} \alpha_j \mathcal{N}(x_i, \mu_j, \sigma_j^2) = \frac{p(y_i = j) \cdot p(x_i|y = j)}{p(x_i)} = p(y_i = j|x_i) \equiv w_{ij}$$

קיבלנו למשה את הפוטורי y_i (האשכול אליו רצים לשיר את x_i), אך הוא לא נתן אלא הוא חבו. כדי לחשב את המבוקש ננחש ערך התחלתי- θ_0 ובudתו נחשב את y_i , ואז בהינתן y_i נבצע עדכון לפרמטרים – נבחן מהו סט הפרמטרים שסביר בזורה הטובה ביותר את האשכולות שהתקבלו בחישוב $-y_i$. באופן פורמלי שני השלבים מנוסחים כך:

E-step – בהינתן אוסף נקודות x וערך עבור פרמטר θ נחשב את האשכול המתאים לכל נקודה, כלומר כל נקודה x_i תוחמת לאשכול מסוים y_i . עבור כל הנקודות y_i נחשב תוחלת ובזרתנה גדר את הפונקציה $Q(\theta, \theta_0)$, כאשר θ הוא פרמטר חדש ו- θ_0 הוא סט הפרמטרים הנוכחיים:

$$Q(\theta, \theta_0) = \sum_{i=1}^n \sum_{j=1}^k p(y_i = j | x_i; \theta_0) \log p(y_i = j, x_i; \theta) = \sum_{i=1}^n \sum_{j=1}^k w_{ij} \log p(y_i = j, x_i; \theta)$$

$$\sum_{i=1}^n \mathbb{E}_{p(y_i|x_i; \theta_0)} \log p(y_i = j, x_i; \theta)$$

M-step – מחשבים את הפרמטר θ שיביא למינימום את $Q(\theta, \theta_0)$ ואז מעדכנים את θ_0 להחדר:

$$\theta = \arg \max_{\theta} Q(\theta, \theta_0)$$

$$\theta_0 \leftarrow \theta$$

חוזרים על התהליך באופן איטרטיבי עד להתקנסות.

עבור Mixture of Gaussians נוכל לחשב באופן מפורש את הביטויים:

$$Q(\theta, \theta_0) = \sum_{i=1}^n \sum_{j=1}^k w_{ij} \log p(y_i = j, x_i; \theta)$$

$$= \sum_{i=1}^n \sum_{j=1}^k w_{ij} \log p(y_i = j; \theta) + \sum_{i=1}^n \sum_{j=1}^k w_{ij} \log p(x_i | y_i = j; \theta)$$

$$= \sum_{i=1}^n \sum_{j=1}^k w_{ij} \log \alpha_j + \sum_{i=1}^n \sum_{j=1}^k w_{ij} \log \mathcal{N}(\mu_j, \sigma_j^2)$$

$$= \sum_{i=1}^n \sum_{j=1}^k w_{ij} \log \alpha_j - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^k w_{ij} \left(\log \sigma_j^2 + \frac{(x_i - \mu_j)^2}{\sigma_j^2} \right)$$

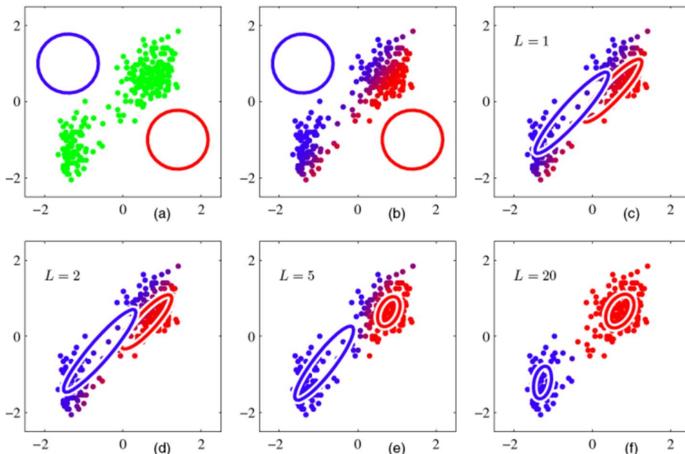
וכעת ניתן לגזר ולמצוא אופטימום:

$$\hat{\alpha}_j = \frac{1}{n} \sum_{i=1}^n w_{ij}$$

$$\hat{\mu}_j = \frac{\sum_{i=1}^n w_{ij} x_i}{\sum_{i=1}^n w_{ij}}$$

$$\hat{\sigma}_j^2 = \frac{\sum_{i=1}^n w_{ij} (x_i - \mu_j)^2}{\sum_{i=1}^n w_{ij}}$$

עבור התפלגויות שונות שאין בהכרח נורמליות יש לחזור לביטוי של $Q(\theta, \theta_0)$ ולבצע עבورو את האלגוריתם.



איור 2.10 20 איטרציות של אלגוריתם EM. מתחילה מינוחש אקראי של ההסתפלוגיות, ובכל איטרציה יש שיפור קר' שההסתפלוגיות מייצגות בצורה יותר טובה את הדadataה המקורי.

ונכון שהאלגוריתם משתפר בכל איטרציה, כאמור בעבר כל (θ, θ_0) מתקיים:

$$\begin{aligned} \log p(x; \theta) &= \sum_y p(y|x; \theta_0) \log p(x; \theta) = \sum_y p(y|x; \theta_0) \frac{\log p(x, y; \theta)}{\log p(y|x; \theta)} \\ &= \sum_y p(y|x; \theta_0) (\log p(x, y; \theta) - \log p(y|x; \theta)) \\ &= \sum_y p(y|x; \theta_0) \log p(x, y; \theta) - p(y|x; \theta_0) \log p(y|x; \theta) \end{aligned}$$

נשים לב שהאיבר הראשון הוא בדיקת $Q(\theta, \theta_0)$. האיבר השני לפני הגדרה הוא האנטרופיה של ההסתפלוגות $p(y|x; \theta_0)$:

$$H(\theta, \theta_0) = - \sum_y p(y|x; \theta_0) \log p(y|x; \theta_0)$$

כעת עברו שני ערכיים שונים של θ מתקיים:

$$\begin{aligned} \log p(x; \theta) - \log p(x; \theta_0) &= Q(\theta, \theta_0) + H(\theta, \theta_0) - Q(\theta_0, \theta_0) - H(\theta_0, \theta_0) \\ &= Q(\theta, \theta_0) - Q(\theta_0, \theta_0) + H(\theta, \theta_0) - H(\theta_0, \theta_0) \end{aligned}$$

לפי א-שיין גיבס מתקיים ($\theta_0, H(\theta, \theta_0) \geq H(\theta_0, \theta_0)$, כלומר):

$$\log p(x; \theta) - \log p(x; \theta_0) \geq Q(\theta, \theta_0) - Q(\theta_0, \theta_0)$$

ולכן עבור כל θ ש מביא לאופטימום את $(\theta_0, H(\theta, \theta_0) - Q(\theta, \theta_0))$, הביטוי $Q(\theta, \theta_0) - Q(\theta_0, \theta_0)$ יהיה חייב' ומילא יהיה שיפור ב- $\log p(x; \theta)$.

2.2.4 Hierarchical Clustering

אלגוריתם נסוף של למידה לא מונחית עבור חלוקת n נקודות ל-K אשכולות נקרא Hierarchical Clustering, והוא מחוליך לשתי שיטות שונות:

ובכך מורדים את מספר אשכולות ב-1, עד שמנגעים ל-K אשכולות. האיחוד בכל שלב נעשה על ידי מציאת שני

האשכולות הקרובים ביותר זה לזה ואיחודם לאשכול אחד. ראשית יש לבחור מטריקה לחישוב מרחק בין שתי נקודות (למשל ורחק אוקלי), מרחק מנהטן ועוד), ולאחר מכן לחשב מרחק בין האשכולות, כאשר יש מספר דרכים להגדיר את המרחק הזה, למשל:

complete-linkage clustering: $\max\{d(a, b) : a \in A, b \in B\}$.

single-linkage clustering: $\min\{d(a, b) : a \in A, b \in B\}$.

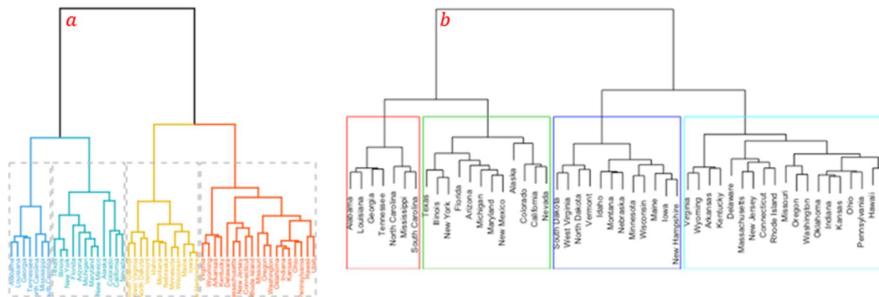
Unweighted average linkage clustering (UPGMA): $\frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} d(a, b)$.

Centroid linkage clustering (UPGMC): $\|c_s - c_t\|$ where c_s, c_t are centroids of clusters s, t, respectively.

עם התקדמות התהילך יש פחת אשכולות, כאשר האשכולות כבר לא מיליכים נקודה אחת בלבד אלא הם הולמים וגדלים. שיטה זו מכונה "bottom up" כיון שהיא תחללה כל נקודה הינה אשכול עצמאי ובכל צעד של האלגוריתם מספור האשכולות קטן באחד. בambilם אחרים, האלגוריתם בניית אשכולות מנצח שבו אין שימוש חלקה לאשכולות לפחות נוצרים אשכולות הולמים וגדלים.

בשיטת זו מבצעים פעולה הפוכה – מסתיכלים על כל הנקודות כאשכול אחד, ואז בכל שלב מבצעים חילקה של אחד האשכולות לפי כל חילקה שנקבע מראש, עד שמייעם ל-K אשכולות. כיון שיש 2^n דרכים לחלק את המdagם, יש הכרח לנתק בשיטות היוריסטיות כדי לקובע את כל החלקה המתאים בכל שלב. שיטה מקובלת לביצוע חילקה נקראת DIANA (DIvisive ANAlysis Clustering) ופיה בכל שלב בוחרים את האשכול בעל השונות היכי גדולה ומחלקים אותו לשניים. שיטה זו מכונה "top-down" כיון שהיא תחללה יש אשכול יחיד ובכל צעד של האלגוריתם מתווסף עוד אשכול.

את התוצאות של האלגוריתם ניתן להראות בצורה נואה באמצעות dendrogram – דיאגרמה הבנויה עץ המציג קשרים בין קבוצות.



איור 2.11 תצוגה של Hierarchical Clustering (a) – התחללה אשכול יחיד ופיזול עד שמגעים למספור האשכולות הרצוי (במקרה זה K = 4). (b) – בהתחילה כל נקודה הינה אשכול, ובכל צעד מחברם שני אשכולות עד שמגעים במספר האשכולות הרצוי.

2.2.5 Local Outlier Factor (LOF)

אלגוריתם Local Outlier Factor (Outliers) הוא אלגוריתם של למידה לא מונחית למציאת נקודות חריגות. האלגוריתם מחשב לכל נקודה ערך הנקרא LOF (Local Outlier Factor), ועל פי ערך זה ניתן לקבוע עד כמה הנקודה היא חלק מקבוצה או לחילוף חריגה ויוצאת דופן.

בשלב ראשון בוחרים ערך k מסוים. עברו כל נקודה x_i , נסמן את k השכנים הקרובים ביותר של x_i – $N_k(x_i)$.icutנ'גדר את k -distance של כל נקודה כמרחק שלמה מהשכן הרחוק ביותר מבין השכנים ב- $N_k(x_i)$. אם למשל $k = 3$ אז $N_k(x_i)$ הוא סט המכיל את שלושת השכנים הקרובים ביותר ל- x_i , וה- k -distance הוא המרחק מהשכן הקרוב אליו הוא קרוב. חישוב המרחק בין שני שכנים נתון לבחירה – זה יכול להיות למשל מרחק אוקלי, מרחק מנהטן ועוד. עבור בחירה של מרחק אוקלי, ניתן להסתכל על k -distance של מעגל המינימלי המכיל את כל הנקודות השוכנות ל- x_i והוא $N_k(x_i)$.

לאחר חישוב ה- k -distance של כל נקודה, מחשבים לכל נקודה Local Reachability Density (LRD) באופן הבא:

$$LRD_k(x_i) = \frac{1}{\sum_{x_j \in N_k(x_i)} \frac{RD(x_i, x_j)}{k}}$$

כאשר $RD(x_i, x_j) = \max(k - \text{distance}(x_i, x_j))$. הגודל LRD מחשב את ההופci של ממוצע המרחקים בין x_i לבין k השכנים הקרובים אליו. ככל שנקודה יותר קרובה ל- k השכנים שלה כך ה-LRD יהיה גדול יותר, ו-LRD קטן משמעו שהנקודה יחסית רוחקה מascalוק הקרוב אליו.

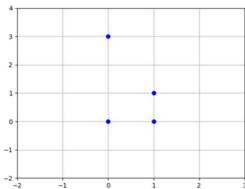
בשלב האחרון בוחנים עבור כל נקודה x_i את היחס בין ה-LRD שלה ו-LRD של $N_k(x_i)$. היחס זה הוא ה-LOF, והוא מחושב באמצעות הנוסחה:

$$LOF_k(x_i) = \frac{\sum_{x_j \in N_k(x_i)} LRD(x_j)}{k} \times \frac{1}{LRD(x_i)}$$

המשמעותי הראשון במכפלה הוא ממוצע ה-LRD של k השכנים של נקודה x_i , ולאחר מכן חישוב הממוצע מחלוקת אותו ב-LRD של הנקודה x_i עצמה. אם הערכאים קרובים, אז ה-LOF יהיה שווה בקירוב ל-1, ואם הנקודה x_i באמת לא שייכת לאשכול של נקודות, אז ה-LOF שלה יהיה נמוך משמעותית מהממוצע של ה-LRD של השכנים שלה, ומילא ה-LOF שלה יהיה גבוה. אם עבור נקודה x_i מתקבל $LOF \approx 1$, אז סביר שהוא חלק מאשכול מסוים.

כדי להמחיש את התהילה' נסתכל על האוסף הבא: $\{A = (0,0), B = (1,0), C = (1,1), D = (0,3)\}$, וקבע $k = 2$. נחשב את ה- k -distance של כל נקודה במונחים של מרחק מנהטן:

$$\begin{aligned} k(A) &= \text{distance}(A, C) = 2 \\ k(B) &= \text{distance}(B, A) = 1 \\ k(C) &= \text{distance}(C, A) = 2 \\ k(D) &= \text{distance}(D, C) = 3 \end{aligned}$$



נחשב את ה-LRD:

$$LRD_2(A) = \frac{1}{\frac{RD(A, B) + RD(A, C)}{k}} = \frac{2}{1+2} = 0.667$$

$$LRD_2(B) = \frac{1}{\frac{RD(B, A) + RD(B, C)}{k}} = \frac{2}{2+2} = 0.5$$

$$LRD_2(C) = \frac{1}{\frac{RD(C, B) + RD(C, A)}{k}} = \frac{2}{1+2} = 0.667$$

$$LRD_2(D) = \frac{1}{\frac{RD(D, A) + RD(D, C)}{k}} = \frac{2}{3+3} = 0.334$$

ולבסוף נחשב את ה-LOF:

$$LOF_2(A) = \frac{LRD_2(B) + LRD_2(C)}{k} \times \frac{1}{LRD_2(A)} = 0.87$$

$$LOF_2(B) = \frac{LRD_2(A) + LRD_2(C)}{k} \times \frac{1}{LRD_2(B)} = 1.334$$

$$LOF_2(C) = \frac{LRD_2(B) + LRD_2(A)}{k} \times \frac{1}{LRD_2(C)} = 0.87$$

$$LOF_2(D) = \frac{LRD_2(A) + LRD_2(C)}{k} \times \frac{1}{LRD_2(D)} = 2$$

כיוון ש- $1 \gg LOF_2(D)$ באופן ייחודי לשאר הנקודות, נסיק כי נקודה D היא outlier.



אior 2.12 – מצאת נקודות חריגות על ידי השוואת ערך LRD של כל נקודה לממוצע LRD של k השכנים שלה. הכל לשה- LOF גודל יותר (העוגל ההפוך), ככל המקרה יותר רוחקה משוכל של נקודות.

יש שני אתגרים מרכזיים בשימוש באלאגוריתם זה – ראשית יש לבחור k יחסית קטן יחסית k מתאים, כאשר k מושפע ממספרם של נקודות רועשות, אך יכול להיות בעייתי במרקם בהם יש הרבה מאוד נקודות חמודות אחת לשנייה, ונקודה שמעט רוחקה מיותרת תזוזה חריגות לмерות שהיא באמתן כן שיכת אלוי. k גדול לעומת זאת יתגבר על בעיה זו, אך הוא לא יזיהה נקודות חריגות שנמצאות בקרוב לאשכולות של נקודות. מלבד אתגר זה, יש צורך לחתת פרשנות למיצאות המתකבלות, ולהחיליט על סך מסוים שרך LOF , שהחל ממנו נקודה מסווגת כחריגה. LOF קwon מ-1 הוא בוודאי לא outlier עבור ערכיו LOF גדולים מ-1 אין כלל חד משמעי עבור איזה ערך הנקודה היא outlier ועבור איזה ערך היא לא. כדי להתמודד עם אתגרים אלו הצענו הרחבות לשיטה המקורית, כמו למשל שימוש בסטטיסטיות שומות המורידות את התלות בבחירה הערך k (LoOP – Local Outlier Probability) או שיטות סטטיסטיות העוזרות לתת פרשנות לערכים המתקבלים (Interpreting and Unifying Outlier Scores).

2.3 Dimensionally Reduction

הוודת ממד (Dimensionality Reduction) הינה טרנספורמציה המפה דאטה לממד נמוך יותר מהממך המקורי, כאשר נרצה שהוודת הממד לא תנסה באופן מהותי את מאפייני הדאטה המקורי. הוודת הממד של דאטה נתון גדרת משתי סיבות עיקריות – אחת טכנית והשנייה מהותית:

א. ביצוע חישובים ופעולות על מערכת ממדים הינה בעלת סיבוכיות גבוהה, ולעתים אף בלתי ניתנת לביצוע.

ב. הוודת הממד של הדאטה קשורה לפחותן מהם המשתנים העיקריים, הפחות חשובים להבנת הדאטה (או שפחות מאפיינים דוגמא נתונה ביחס לזוגיותות אחרות). לעיתים התחשבות במסתנים המשניים משמשה לרעה על ביצוע המודל, למשל על ידי הוספה רעה ולא מדעת. תופעה זו נקראת קלلت הממדיות (curse of dimensionality). יתרון נוסף של הוודת ממד טמון בויזואלייזציה של המידע, כך שנitin להציג על ידי 2 או 3 ממדים עיקריים, בעוד גրף דו-ממדי או תלת-ממדי בהסתrema.

דוגמה למערכת מרובת ממדים יכולה להיות מדידת רמות חלבונים (פרוטואינים) של גנים (genes) המבוטאים בתא חי, כאשר כל ממד, או מאפיין (פייצ'ר), מהתאים לאן אחר. באופן כללי, יתכן ונמדדים בכל ניסוי מאות תאים, כאשר לכל תא גמדדות רמות ביוטי של מאות או אלפי גנים. כמות עצמה זו של מידע בממד גבוה (אלפי תאים ואלפי גנים בכל תא) מאיירתת את המחבר – הן מבחינת זיהוי המאפיינים, או רמות האגנים המבוטאים, הרלוונטיים ומשפיעים ביותר, והן מבחינת ניסיון למלול את הדאטה בצורה כמה שיותר פשוטה. במחקר משנת 2007 נלקחו 105 דימות של תא סרטן שד, כאשר לכל דגימה (או דוגמא) נמדדו רמות התבניות של 27,648 גנים שונים. כמון שלנתה את המידע בצורה הגלומית זו משימה בלתי אפשרית, ויש הכרח לבצע עליה מניפולציה כלהיא כדי שהיא אפשר לעבד אותו.

ישן שיטות מרובות לביצוע הורדת ממד לדאטה, כאשר ניתן לסייע לשתי קטגוריות עיקריות: בחירת מאפיינים (feature selection), והטלת מאפיינים (projection). השיטות השניות לקטגוריה הראשונה מנוסות לבחור את המאפיינים (המשתנים) המתאימים ביותר ממדע הנתן. שיטות מהקטורייה השנייה, המתוארכות בפרק זה, נוקטות בגיןה של הטלה, טרנספורמציה, של המאפיינים הקיימים לייצוג על ידי סט של מאפיינים חדשים במרחב אחר (ולרוב פשוט יותר). חשוב להציג בשיטות מהקטgorיה הראשונה, מאפיינים חדשים רלוונטיים מושתמים. בגיןו לכך, בשיטות שנדרן בהן בפרק זה, המבוססות על הטלת המאפיינים, כל מאפיין חדש הינו טרנספורמציה של כל האחרים, ולא רק של חלקי. כך, המאפיינים החדשניים מילאים, או לוקחים בחשבון, כל אחד מהמאפיינים הננדדים המקוריים, ללא השמטה.

ניתן לבצע הטלת מאפיינים באמצעות טרנספורמציות ליניארית או לא-LINIARITY. בפרק זה עוסק בטרנספורמציה ליניארית אחת, הנקראת ניתוח גורמים ראשיים (Principle Component Analysis) ונתשי טרנספורמציות לא-LINIARITY (t-SNE, UMAP). נציין כי קיימות עוד טרנספורמציות, ליניארית ולא-LINIARITY, המשמשות להורדת ממד של דאטה שאין יזכיר כאן.

2.3.1 Principal Components Analysis (PCA)

כפי שהזכר לעיל, ניתוח גורמים ראשיים מבוסס על טרנספורמציה ליניארית של המאפיינים הקיימים. המטרה של אלגוריתם זה היא לבנות "יצוג חדש ובעל ממד נמוך יותר מאשר הממד המקורי, מתקרטה לשמר כמה שיטות את השונות של המאפיינים (features) של הדאטא המקורי. מדוע השונות כה משמעותית? ניקח למשל מאפיין המופיע בכל הדוגמאות בדאטא הנתון, ובכלן מאפיין זה הוא בעל אותו ערך. מאפיין זה הוא בעל שונות אפס, ולמעשה הוא לא מכל שום מידע על הדאטא ונרצה להפטר ממנו. בדומה לכך, אם יש שני מאפיינים בעלי תלות ליניארית – אין סмы לשמר את שנייהם כיוון שידיית האחד מאפשרת לדעת בס את השני. במקורה זה הקורלציה בין שני המאפיינים היא 1, והשתרת המאפיין השני לא תתרום "לשונות הכלולות" של "יצוג הדאטא". נראתה בהמשך שייצוג שבנה באמצעות PCA לא יכול פיצרים כללו. עד נעיר כי האלגוריתם דואג לכך שהמודלים של כל צירוף ליניארי יהיו וקטורים בעלי אורך של 1, בכך לא לנפח באופן מלאכותי את השונות של המאפיינים החדשניים (לאחר הורדת ממד).

לאחר הקדמה זו, נפרט כיצד מחושבים המאפיינים החדשניים, הנקראים "הגורמים העיקריים". הגורם הראשון (first principal component, PCA₁) היה הצירוף הליניארי של המאפיינים המקוריים בעל השונות האגדולה ביותר. הגורם השני השני (second principle component, PCA₂) הוא גם צירוף ליניארי של המאפיינים המקוריים, השונות שלו היא השניה האגדולה ביותר, ובנוסף דורשים שהוא אורתוגונלי ל-PCA₁: PCA₁ ⊥ PCA₂. הגורם השלישי, הוא צירוף ליניארי בעל השונות השלישית האגדולה ביותר, ומואזן לשני הגורמים המקוריים – PCA₁ ⊥ PCA₂ ⊥ PCA₃. וכך הלאה כך שהגורם הראשון מסדר i , הוא בעל השונות ה- i -ית האגדולה ביותר תחת אילוץ של גורמים מאונכים – $\sum_j \text{PCA}_j = 0$. הורדת הממד מתבצעת על ידי ליקית מסגור גורמים ראשונים וראשונים, והזנתה הקטנים ביותר.

לאחר שאפיינו את הגורמים הראשיים בהם הם מעוניינים, עליה השאלה כיצד לבצע טרנספורמציה ליניארית שבעזרתה ניתן למצאו את הגורמים הראשיים האלה. נניח שבידינו דאטה $\mathcal{X} \in \mathbb{R}^{M \times N}$, הכולמת M דוגמאות שונות, שכל אחת מהן היא בלוטת N מאפיינים [למשל, עבור הדוגמא של תא סרטן השד, נתנו מידע- M = 105-ה-תאים שונים, כאשר עבור כל תא נמדד רמות ביוטי של N גנים שונים]. מטריצה זו לעתים נקראת הדאטא, design matrix של הדאטא, ונסמן אותה באופן הבא:

$$\hat{\mathcal{X}} = \begin{bmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_M \end{bmatrix} = [\vec{x}^1, \dots, \vec{x}^N] \in \mathbb{R}^{M \times N}$$

כאשר כל וקטור שורה, $\vec{x}_m, m \in \{1, \dots, M\}$, הינו נתוני המדידות של המאפיינים השונים בדוגמא מס' m , ובהתאם, וקטור עמודה $\vec{x}^n, n \in \{1, \dots, N\}$ (שימו לב לשינוי סימן, אינדקס עליון עבור וקטור עמודה), הינו נתוני המדידות של מאפיין מסוים על כל הדוגמאות. נניח שסכום המדידות עבור כל מאפיין הוא אפס, זאת אומרת לכל מאפיין i -י מתקיים:

$$\text{mean}(\vec{x}^n) = \sum_{m=1}^M x_{m,n} = \vec{0}$$

מכיוון שכל עמודה של המטריצה מסוימת ערכים של מאפיין מסוים במדידות שונות, סכום כל עמודה במטריצה \hat{X} הוא אפס.icut, נרצה לבצע הטלה (טראנספורמציה) לינארית, זאת אומרת נכפיל את מטריצה \hat{X} במטריצה משקלים \hat{W} :

$$\hat{T} = \hat{X} \cdot \hat{W}$$

אם נסמן את השורה ה- m -ית במטריצה \hat{T} על ידי \vec{T}_m , נקבל:

$$\vec{T}_m = \vec{X}_m \cdot \hat{W}$$

כאשר המטריצה $\hat{W} \in \mathbb{R}^{N \times K}$, כך ש- $\hat{T} \in \mathbb{R}^{M \times K}$. הטלה זו מביאה לכך שלאחר הטרנספורמציה נשארים רק K מאפיינים. כיוון שאנו מעוניינים בהורדת הממד, קרי הורדת מספר המאפיינים, נדרש $N \leq K$.

את תהליך מציאת מטריצת המשקלים ניתן לנסח באופן פורמלי על ידי שלושה תנאים:

1) כל עמודה של מטריצת המשקלים תהיה בעלת נורמה השווה ל-1, ככלומר:

$$\|\hat{W}^k\|^2 = \sum_{m=1}^M (W_{m,k})^2 = 1$$

2) השונות עבור המאפיין k -י, המוגדרת על ידי $s_k^2 = (\vec{T}^k)^T \vec{T}^k = \sum_{m=1}^M (T_{mk})^2$, מקיימת:

3) העמודות של \hat{W} אורtagonalיות זו לזו, זאת אומרת $\hat{W}^k \perp \hat{W}^l$ לכל שתי עמודות k, l .

נראה זאת באופן מפורש: נתחיל במצבה הראשונה \hat{W}^1 . נדרש:

$$\hat{W}^1 = \underset{\|\hat{W}\|=1}{\operatorname{argmax}}(s_1^2)$$

זאת אומרת:

$$\begin{aligned} \hat{W}_1 &= \underset{\|\hat{W}\|=1}{\operatorname{argmax}}(s_1^2) = \underset{\|\hat{W}\|=1}{\operatorname{argmax}}\left(\left(\vec{T}^1\right)^T \cdot \vec{T}^1\right) = \underset{\|\hat{W}\|=1}{\operatorname{argmax}}\left(\left(\hat{X} \hat{W}^1\right)^T \cdot \hat{X} \hat{W}^1\right) \\ &= \underset{\|\hat{W}\|=1}{\operatorname{argmax}}\left(\left(\hat{W}^1\right)^T \left(\hat{X}\right)^T \cdot \hat{X} \hat{W}^1\right) \end{aligned}$$

ולכן העמודה הראשונה של מטריצת המשקלים \hat{W}^1 נתונה על ידי:

$$\hat{W}^1 = \underset{\|\hat{W}\|=1}{\operatorname{argmax}}\left((\hat{W}^1)^T \cdot \hat{S} \cdot \hat{W}^1\right)$$

כאשר מטריצה $\hat{S} \in \mathbb{R}^{(N \times N)}$ הינה מטריצת השונות המשותפת (covariance), המוגדרת על ידי $\hat{S}_{ij} = (\hat{X})^T \cdot \hat{X}$. מטריצה זו, מסדר $N \times N$, מדירה את השונות המשותפת בין צמד מאפיינים, כאשר $\hat{S}_{\nu_1, \nu_2} = \sum_{m=1}^M X_{\nu_1, m} X_{\nu_2, m}^T$. ניתן לשים לב כי מטריצה זו סימטרית וממשית (ולכן הרミיטית).

לפי משפט המינימום-מקסימום (קורנט-פישר-ויל, מובא כנספח לפרקי): עבור \hat{S} מטריצה הרמייטית $(S_{ij} = S_{ji}^*)$ בעלת ערכים עצמיים $\lambda_1, \lambda_2, \dots, \lambda_K$, מתקיים:

$$\lambda_1 = \max_{\|\hat{W}\|=1}\left((\hat{W}^1)^T \cdot \hat{S} \cdot \hat{W}^1\right)$$

כאשר \hat{W}^1 הינו הווקטור העצמי המתאים לערך העצמי המקסימלי של \hat{S} : λ_1 .

icut, כדי למצוא את הווקטור העצמי הבא, \hat{W}^2 , והערך העצמי המתאים לו λ_2 , נגדיר מטריצה חדשה \tilde{X} :

$$\tilde{X} = \hat{X} - \hat{X} \hat{W}^1 (\hat{W}^1)^T$$

$$\begin{aligned}
\hat{W}^2 &= \underset{\|\hat{W}\|=1}{\operatorname{argmax}}(s_2^2) = \underset{\|\hat{W}\|=1}{\operatorname{argmax}}\left(\left(\vec{T}^2\right)^T \cdot \vec{T}^2\right) \\
&= \underset{\|\hat{W}\|=1}{\operatorname{argmax}}\left(\left(\hat{W}^2\right)^T\left(\tilde{X}+\hat{X}\hat{W}^1\left(\hat{W}^1\right)^T\right)^T \cdot\left(\tilde{X}+\hat{X}\hat{W}^1\left(\hat{W}^1\right)^T\right) \hat{W}^2\right) \\
&= \underset{\|\hat{W}\|=1}{\operatorname{argmax}}\left(\left(\hat{W}^2\right)^T\left(\tilde{X}\right)^T \cdot\left(\tilde{X}\right) \hat{W}^2\right)
\end{aligned}$$

כאשר \hat{W}^2 הינו הווקטור העצמי המתאים לערך העצמי המקסימלי של $(\tilde{X})^T$, ובעצם הוא הערך העצמי השני בגודלו עבור מטריצה $\hat{X}^T\hat{X}=\hat{S}$. (בчисלוב השימושנו בשיטה כ- $\hat{W}^2 \perp \hat{W}^1$).

באופן כללי, כדי למצוא את \hat{W}^k והערך העצמי המתאים לו λ_k , נגדיר מטריצה חדשה \tilde{X} באופן הבא:

$$\begin{aligned}
\tilde{X} &= \hat{X}-\sum_{i=1}^{k-1} \hat{X} \hat{W}^i\left(\hat{W}^i\right)^T \\
\hat{W}^k &= \underset{\|\hat{W}\|=1}{\operatorname{argmax}}\left(\left(\hat{W}^k\right)^T\left(\tilde{X}\right)^T \cdot\left(\tilde{X}\right) \hat{W}^k\right)
\end{aligned}$$

כך λ_k הינו הערך העצמי המקסימלי k -י של מטריצת השונות המשותפת $\hat{X}^T\hat{X}=\hat{S}$.

ניתן גם, באופן פשוט יותר, להשתמש בשיטת פירוק ערכאים סינגולריים (SVD), כאשר נמצא את הפירוק המתאים למטריצת השונות המשותפת:

$$\hat{S}=\hat{W} \cdot \hat{\Lambda} \cdot \hat{W}^T$$

כאשר $\hat{\Lambda}$ הינה מטריצה אלכסונית, ו- $i_{ii}=\lambda_{ii}$ הינם הערכים העצמיים של \hat{S} המסודרים לפי גודלם מהגדול לקטן - $\lambda_M \geq \lambda_2 \geq \lambda_1 \geq \dots \geq \lambda_k$, ומטריצה \hat{W} מורכבת מוקטוריו عمودה שהיבנים הווקטורים העצמיים המתאימים לערכאים העצמיים. הווקטורים העצמיים בהגדרתם הינם אורתוגונליים זה לזה, וכיון ש- $1=\|\hat{W}^k\|$ לכל k , הם בעצם אורתונורמליים.

לסיכום, על מנת למצוא את הגורמים הראשיים עבור המידע הנוכחי \hat{X} :

- "מרכז" את הנתונים כך שהממוצע עברו כל מאפיין הוא אפס: $\hat{X}^m=\hat{X}^m-\operatorname{mean}_n(\hat{X}^m)$
- מציא את מטריצת השונות המשותפת $\hat{S}=\left(\hat{X}\right)^T \cdot \hat{X}$
- מציא את $\hat{W}^T \cdot \hat{W}=\hat{S}$ – ה-SVD של \hat{S} .
- חישוב $\hat{W}=\hat{S} \cdot \hat{X}$.

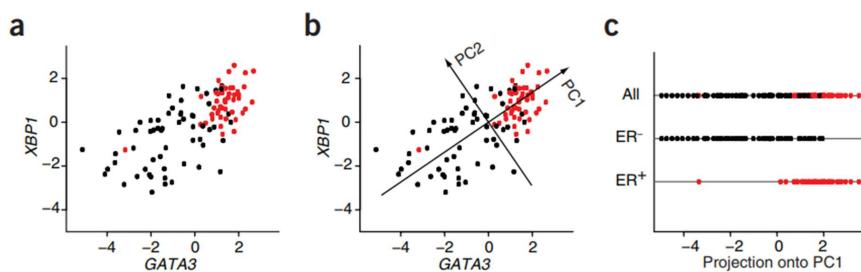
הגורמים הראשיים נתונים על ידי $\hat{W}^m=\hat{X}^m \hat{W}$ ו- \hat{T}^m .

נציין שלשיטות הנитוח של גורמים ראשיים יש מספר מגבלות. ראשית, היא נונטנת "משקל יתר" על מאפיינים שהשווות בהם גודליה, ללא קשר לחישובתם, או ליחידות שביהן המאפיין נמדד (זאת אונרמת לדוגמא שלגביה שמדד בסוניים מיטיים יונן "משקל" גובה יותר מאשר גובה הנמדד במטרים). שנית, שיטתית זו מחייבת כי הממד החשוב הוא השונות המשותפת שהוא בעצם קורלציה ליניארית בין שני משתנים, אולם "תיכון" במערכות מסוימות שדואוקה הקורלציה הלא-LINEARית היא החשובה יותר. כמו כן, לעיתים "מרכז" המידע גורם לתוצאות לאבד ממשמעותן.

כדי להתגבר על המגבלות בשיטת PCA שהציגו לעיל פוטחו שיטות נוספות או משלימות. לדוגמה, ניתן למצער את השפעת יחידות המידה על המאפיינים על ידי פיצכתם לחסרי יחידות. בנוסף, יש שיטות הלוקחות בחשבון קורלציות לא-LINEARיות, לדוגמא שיטת kernel PCA, או שיטות להתחדשות עם בעיית המרcco על ידי דרישת משתנים חיוביים (NMF).

לצורך המchéשה של תהליכי חישוב הגורמים העיקריים ניתן שתי דוגמאות. ראשית נזכיר שהזכרנו בתחילת פרק זה – מחקר שפורסם בשנת 2007 ובו נלקחו 105 דגימות של תא סרטן אחד, כאשר לכל דימה ממדדי רמות התבניות של 27,648 גנים שונים. לשם הדגמה, נשתמש בניתוח שפורסם כונה לאחר מכן (ב-2008) על ידי אחד מעורכי המחבר המוביל. שם, החוקר מציג רמות של שני חלבונים; האחד בשם GATA3, והשני בשם XBP1, כאשר

דגימות תא הסרטן מסווגות לפי סוג קולטני האסטログן שלהם (+ או -). כתם, על ידי "סיבוב" מערכת הצירים בעדרת טונספורמציה לינארית PCA כפי שהסביר לעיל – נמצא כי ניתן לסוג, ללא איבוד מידע רב, את מצב קולטני האסטログן בתאי סרין השד על ידי הגורם הראשי הראשון PCA₁, כפי שניתן לראות באירוע. יש לשים לב שהגורם הראשי הראשון PCA₁, מכיל מידע משני החלבוניים.



איור 2.13 (a) רמות ביוטי של שני חלבונים (X ו-XBP1) (ציר ה-X) ו-GATA3 (ציר ה-Y). קולטני אסטרוגן חוביים או שליליים מסווגים באדום ושחורים בהתאם. (ב) מציאת הגורמים הראשיים, וסיבוב מערכת הצירים. בהתאם לתיאוריה, ניתן להבחין כי השונות של המידע על גבי הציר החדש PCA₁ אינה מתקבלת כפוקיזמייה. (ג) הציג תוצאות הדידיה כפוקיזמייה של PCA₁ בלבד. בגרף זה ניתן לראות כי בירור כיצד חורדת הממד מסייעת למציאו הבנה פשוטה (בממד אחד) בין קולטני האסטרוגן.

נזכיר לבסוף דוגמא חשובה מפורטת. נניח וננתן המערכת הדו-ממדית הבאה:

$$X = \begin{pmatrix} -0.5 & -0.4 \\ -0.4 & -0.1 \\ 0.1 & 0 \\ 0.3 & 0.3 \\ 0.5 & 0.2 \end{pmatrix}$$

מערכת הנתונים מכיל 5 דוגמאות, וכל דוגמא נמדדדי שני מאפיינים ($2 = M, N = 5$). שורות המטריצה מציגות את המדידות השונות, והעמודות מייצגות את מאפיינים.

מערכת כבר ממורכז, כלומר המאפיין הראשי מקיים:

$$\text{mean}_1(X^m) = \sum_{m=1}^5 X_{m1} = -0.5 - 0.4 + 0.1 + 0.3 + 0.5 = 0$$

ועבור המאפיין השני:

$$\text{mean}_2(X^m) = \sum_{m=1}^5 X_{m2} = -0.4 - 0.1 + 0 + 0.3 + 0.2 = 0$$

ນוחש את מטריצת השונות המשותפת:

$$\begin{aligned} S &= (\hat{X})^T \hat{X} = \begin{pmatrix} -0.5 & -0.4 & 0.1 & 0.3 & 0.5 \\ -0.4 & -0.1 & 0 & 0.3 & 0.2 \end{pmatrix} \begin{pmatrix} -0.5 & -0.4 \\ -0.4 & -0.1 \\ 0.1 & 0 \\ 0.3 & 0.3 \\ 0.5 & 0.2 \end{pmatrix} \\ &= \begin{pmatrix} 0.5^2 + 0.4^2 + 0.1^2 + 0.3^2 + 0.5^2 & 0.5 \cdot 0.4 + 0.4 \cdot 0.1 + 0.1 \cdot 0 + 0.3^2 + 0.5 \cdot 0.2 \\ 0.5 \cdot 0.4 + 0.4 \cdot 0.1 + 0.1 \cdot 0 + 0.3^2 + 0.5 \cdot 0.2 & 0.4^2 + 0.1^2 + 0^2 + 0.3^2 + 0.2^2 \end{pmatrix} \\ &= \begin{pmatrix} 0.76 & 0.43 \\ 0.43 & 0.3 \end{pmatrix} \end{aligned}$$

על מנת ללקסן מטריצה זו, נמצא את הערכים העצמיים של המהוים שורשים של המשוואה האופיינית הבאה:

$$0 = |\hat{S} - \lambda I| = \begin{vmatrix} 0.76 - \lambda & 0.43 \\ 0.43 & 0.3 - \lambda \end{vmatrix} = (0.76 - \lambda)(0.3 - \lambda) - 0.43^2 \approx (\lambda - 1.02)(\lambda - 0.04)$$

כאשר לפולינום אופייני זה שני שורשים: $\lambda_1 \approx 1.02, \lambda_2 \approx 0.04$ (יש לשים לב שבחרנו $\lambda_1 > \lambda_2$).
נמצא כעת את הווקטור העצמי המתאים לערך העצמי הגדל מבין השניים – λ_1 . וקטור זה, המסומן עלי ידי \hat{W}^1 ,
מקיים:

$$\hat{S}\hat{W}^1 = \lambda_1\hat{W}^1$$

כך ש:

$$0 = (\hat{S} - \lambda_1\hat{I})\hat{W}^1 \approx \begin{pmatrix} -0.83 & 0.107 \\ 0.107 & -0.94 \end{pmatrix} \begin{pmatrix} W_{11} \\ W_{21} \end{pmatrix} \Rightarrow \hat{W}^1 \approx \begin{pmatrix} 0.86 \\ 0.51 \end{pmatrix}$$

הווקטור העצמי השני, \hat{W}^2 , המתאים לערך העצמי השני אופן, ומתקיים:

כך שמטריצת המשקלים נתונה על ידי:

$$\hat{W} = (\hat{W}^1 \quad \hat{W}^2) = \begin{pmatrix} 0.86 & 0.51 \\ 0.51 & -0.86 \end{pmatrix}$$

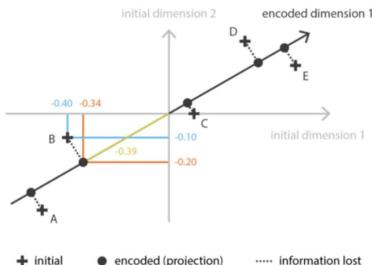
הטלת המדידות למערכת המאפיינים החדשה נתונה על ידי:

$$\hat{T} = \hat{X} \cdot \hat{W}$$

לכן, המדידות של הגורם הראשי הראשון, נתונות על ידי

$$\hat{T}^1 = \hat{X} \cdot \hat{W}^1 \approx \begin{pmatrix} -0.5 & -0.4 \\ -0.4 & -0.1 \\ 0.1 & 0 \\ 0.3 & 0.3 \\ 0.5 & 0.2 \end{pmatrix} \begin{pmatrix} 0.86 \\ 0.51 \end{pmatrix} = \begin{pmatrix} -0.5 \cdot 0.86 - 0.4 \cdot 0.51 \\ -0.4 \cdot 0.86 - 0.1 \cdot 0.51 \\ 0.1 \cdot 0.86 \\ 0.3 \cdot 0.86 + 0.51 \cdot 0.3 \\ 0.5 \cdot 0.86 + 0.2 \cdot 0.51 \end{pmatrix} \approx \begin{pmatrix} -0.63 \\ -0.39 \\ 0.09 \\ 0.41 \\ 0.53 \end{pmatrix}$$

נראה זאת באופן גרפי:



אייר 2.14 הורדת מדד של דатаה דו-ממדי לממד אחד.

נספח: משפט המינימום-מקסימום (גורנט-פישר-וייל)

עבור $\hat{S} \in \mathbb{R}^{M \times M}$ מטריצה הרミיטית ($S_{ij} = S_{ji}^*$) מסדר M עם ערכים עצמיים $\lambda_1, \lambda_2, \dots, \lambda_M$ מתקיים:

$$\begin{aligned} \lambda_m &= \min_U \left\{ \max_{\substack{x \in U, \\ \|x\|=1}} \{x^\dagger S^\dagger x \mid x \in U, x \neq 0\} \middle| \dim(U) = M - m + 1 \right\} \\ &= \min_U \left\{ \max_{\substack{x \in U, \\ \|x\|=1}} \left\{ \frac{x^\dagger S^\dagger x}{x^\dagger x} \right\} \mid x \in U, x \neq 0 \right\} \mid \dim(U) = M - m + 1 \end{aligned}$$

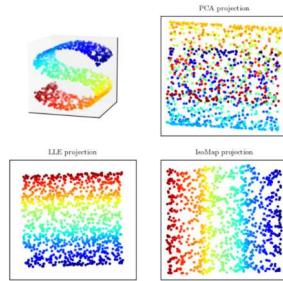
הערך העצמי המקסימלי מקיים:

$$\lambda_1 = \max_{\|W\|=1} ((\hat{W}^1)^T \cdot \hat{S} \cdot \hat{W}^1)$$

כאשר \hat{W} , הינו הערך העצמי המותאים ל- λ_1 – ערך העצמי המקסימלי של $\hat{\Lambda}$.

2.3.2 t-distributed Stochastic Neighbors Embedding (t-SNE)

אלגוריתם הורדת הממד PCA פועל באופן יינארי, מה שמקל על תהליך החישוב שלו, אך מגביל את יכולות ההכללה שלו. אלגוריתם אחר, לא יינארי, נקרא t-SNE-ט, והוא מנשה לקחת את הדעתה בממד גובה ובמציאות שימוש בכלים סטטיסטיים למפות אותו למערכת דו-ממדית או תלת-ממדית.



איור 2.15 הדגמה של מגבלת שיטת PCA ויתרונות השיטות הלא-ליינאריות להורדת ממד. באIOR השמאלי העליון מוצגת דאטא תלת ממד בעל מבנה צבירות האות S . בנצח, לדאטא ישנו מאפיין מסווג המתבטא בעקבות שניים ובמעבר שלמה מכחול לאדם. טונומסומציה לינארית PCA (סיבוב/מטריצה) אינה מצליחה לבנות הדעתה הגדריאנס בעמאותו לפחות (ראה איור עליון ימינו). אולם, שיטות לא-ליינאריות יכולות מצליחות להוריד את הממדיות של הבעה ולשמור את מבנה היריעה הסטולוגית (ראה שי אירום תחתונים).

לשם כך, נשתמש באותו מערך נתונים, $\vec{X} \in \mathbb{R}^{M \times N}$, כאשר M הוא מספר הדוגמאות, ו- N הוא מספר המאפיינים (\vec{x}_i המשותנים/ \vec{p}_i 'ים). חשוב לשים לב כי כל מדידה מוצגת על ידי וקטור שורה \vec{x}_m . הרעיון הכללי של השיטה הוא למפות את סט המדידות באופן כזה שמדידות דומות יותר, קרי מדידות "קרובות" יותר במרחב ה- N -ממדי, יוצגו על ידי נקודות קרובות יותר במרחב חדש K -ממד, כאשר לרובה $3 \leq K \leq N$. נסמן את המרחב המקורי \mathcal{X} ואת המרחב החדש ב- \mathcal{Y} , כאשר בשני המרחבים המדידות מוצגות על ידי נקודות בגרף פיזור (scatter plot). המטריקה המשמשת למדידת דמיון (similarity) בין שתי נקודות במרחב המקורי \mathcal{X} הינה הסתברותית. עברו שתי מדידות m_1, m_2 במרחב המקורי ה- N -ממדי, ההתפלגות הנורמלית המשותפת P_{m_1, m_2} הינה:

$$P_{m_1, m_2} = \frac{Z_1^{-1}}{2N} \exp\left(-\frac{\|\vec{x}_{m_1} - \vec{x}_{m_2}\|^2}{2\sigma_1^2}\right) + \frac{Z_2^{-1}}{2N} \exp\left(-\frac{\|\vec{x}_{m_1} - \vec{x}_{m_2}\|^2}{2\sigma_2^2}\right)$$

כאשר σ_i נקרא perplexity (perplexity) והוא פרמטר שנקבע מראש, ו- Z_i הינו קבוע המormalיזציה, המוגדר על ידי $Z_i = \sum_{k \neq i} \exp\left(-\frac{\|\vec{x}_i - \vec{x}_k\|^2}{2\sigma_i^2}\right)$. עבור נקודות קרובות יותר, עבור הביטוי $\|\vec{x}_{m_1} - \vec{x}_{m_2}\|$ קטן, ההסתברות שהנקודה $\vec{x}_{m_1} - \vec{x}_{m_2}$ שכנה של \vec{x}_{m_2} גדולה. לעומת זאת עבור נקודות רחוקות זו מזו, ככלור $\|\vec{x}_{m_1} - \vec{x}_{m_2}\|$ גדול, ההסתברות שהנקודה \vec{x}_{m_1} שכנה של \vec{x}_{m_2} קטנה מאוד.

icut, כפי שהוזכר לעיל, נרצה למפות את סט המדידות: $\begin{bmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_M \end{bmatrix} \rightarrow \begin{bmatrix} \vec{y}_1 \\ \vdots \\ \vec{y}_M \end{bmatrix}$, כך שהמדד של \vec{y}_m הינו נמוך (2 או 3 ממדים). בנצח, נדרש שנקודות דומות ("שכנות") במרחב \mathcal{X} , ישארו שכנות לאחר המיפוי למרחב \mathcal{Y} . מתרבר שפונקציית הסתברות המותנית, המתחילה לתיירור דמיון בין נקודות שכנות במרחב החדש \mathcal{Y} , הינה הבלתי-ט, הנקראת גם התפלגות סטודנט עם דרגת חופש אחת (דזון בבעין) לבחור בפונקציות הסתברות אלו בהמשך). קר, נכמת את הדמיון בין m_1 לבין m_2 , על ידי ההסתברות המשותפת Q_{m_1, m_2} המוגדרת באופן הבא:

$$Q_{m_1, m_2} = 3^{-1} \frac{1}{1 + \|\vec{y}_{m_1} - \vec{y}_{m_2}\|^2}$$

$$\text{כאשר } \sum_{k \neq j} \left(1 + \|\vec{Y}_k - \vec{Y}_j\|^2 \right)^{-1} = 3.$$

המיופיע בין מרחיב המקורי \mathcal{X} לבין המרחיב החדש \mathcal{Y} הוא מיטבי אם הוא "משמר" את השכנות של נקודות (מדידות) קרובות. לשם כך נגידר את פונקציית המחר על ידי Kullback-Leibler divergence, הבונה מרחיק בין שתי התפלגות:

$$C = \mathcal{D}_{KL}(P|Q) \equiv \sum_{m_1} \sum_{m_2} P_{m_1, m_2} \log \left(\frac{P_{m_1, m_2}}{Q_{m_1, m_2}} \right)$$

עבורו פונקציית המחר מינימלית, ולשם כך השתמש בגרדיאנט לפי \vec{Y}_{m_i} :

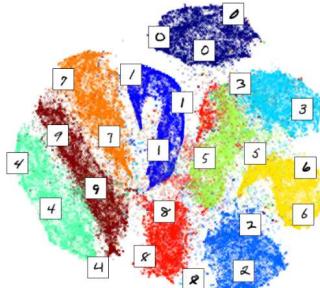
$$\begin{aligned} \frac{\delta C}{\delta \vec{Y}_{m_i}} &= \frac{\delta}{\delta \vec{Y}_{m_i}} \left[\sum_{m_1} P_{m_1, m_i} \log \left(\frac{P_{m_1, m_i}}{Q_{m_1, m_i}} \right) + \sum_{m_2} P_{m_1, m_2} \log \left(\frac{P_{m_1, m_2}}{Q_{m_1, m_2}} \right) \right] \\ &= 4 \sum_{m_1} (P_{m_1, m_i} - Q_{m_1, m_i}) \left(1 + \|\vec{Y}_{m_1} - \vec{Y}_{m_i}\|^2 \right)^{-1} (\vec{Y}_{m_1} - \vec{Y}_{m_i}) \end{aligned}$$

חישוב המינימום באמצעות אנלטי לא תמיד אפשרי או לא תמיד יעיל, ולכן מקובל להשתמש בשיטת gradient descent שינהה שיטה איטרטיבית למציאת נקודת המינימום של פונקציה (פירוט על שיטה זו וריאציות שונות שלה מופיע בפרק 4.3.5). עבור הורדת המנד, חישוב המינימום בעדרת שיטה זו עשויה באופן הבא:

- א. אתחלו: * נתון $\in \mathbb{R}^{M \times N} X$.
- * היפר-פרמטר לפונקציית הדמיון: בחירת השונות σ .
- * בחירת שיטת אופטימיזציה (למשל: ADAM, SGD), והיפר פרמטרים כגון: קצב הלמידה ϵ , מומנטום (t) , גודל ה-batch וכו'.
- ב. חשב את P_{m_1, m_2} .
- ג. אתחל את המייפוי $(\hat{I}_M) = \{\vec{Y}_1, \vec{Y}_2, \dots, \vec{Y}_M\} \sim N(0, s^2 I_M)$ [ז"א בחר את הערכים ההתחלתיים לפי התפלגות גאוסיאנית עם ממוצע 0 וסטיית תקן s (בבחר להיות קטן, נניח $s = 10^{-4}$). I_M מטריצת יחידה].
- ד. עברו איטרציה t :
 - * חשב את Q_{m_1, m_2} עבור ה-batch.
 - * חשב את הגרדיאנט של פונקציית המחר.
 - * עדכן: $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) [\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}]$

נעיר כי בשפות תכנות רבות, האלגוריתם עצמו כבר מוגדר על ידי פונקציות מובנות, ויש רק להגדיר את הפרמטרים הדרושים.

במאמר המקורי שהציג את השיטה הובאה דוגמא של שימוש באלגוריתם עבור הטלה של הספורות 0 עד 9, המיצגות על ידי תמונות בממד גובה $\mathbb{R}^{28 \times 28}$, למרחב דו-ממדי. בדוגמה זו נלקחו 6,000 תמונות של ספורות ומיפוי אותן למרחב דו-ממדי. במרחב זה ניתן לראות בברור כיצד כל תמונה מופתת לאחר אחר, כיוון שבפועל נוצרו עשרה אשכולות שוות, המוגhanim彼此 בחרואה אחד מהשנים. ביצוג הדו-ממדי אין משמעות לציריהם, כיוון שבאלגוריתם זה יש חישבות רק למרחק היחסי בין הנקודות.



איור 2.15 הציגו (יזואלייזציה) דו-ממדית של מערך נתונים עבור כתבי-יד של ספרות (MNIST) על ידי שיטת t-SNE. כל דוגמא \vec{X}_m מאופיינת על ידי $28 \times 28 = 784$ ערכים (פיקסלים בגוון אפור) ומוסוגת בהתאם למספרה בין 0 ל-9. באירוע מוגנות 6000 נקודות (מידיות) כללו, כאשר צבעים שונים מייצגים ספרות שונות. מלבד הבחינה בין הספרות, ניתן לראות שספרות דומות קרובות זו לזו גם במרחב חדש (למשל הספרה 1 קרויה לספרה 7, שבturnora קרויה לספרה 9).

כאמור, פונקציית הדמיון בין שתי נקודות במרחב המקורי הינה הפילוג הנורמלי המשותף של שתי הנקודות, ואילו במרחב החדש פונקציית הדמיון הינה התפלגות t . שתי העוררות הקשורות על ידי:

a. סימטריה:

פונקציית הדמיון האגואיאנית בין שתי נקודות במרחב X הינה פונקציה סימטרית, כלומר $P_{m_1, m_2} = P_{m_2, m_1}$. אולם ניתן להגדיר גם פונקציית דמיון א-סימטרית, המבוססת על התפלגות מותנית (במקום התפלגות משותפת). הפונקציה המותנית נתונה על ידי:

$$P_{m_1|m_2} = Z_2^{-1} \exp\left(-\frac{\|\vec{X}_{m_1} - \vec{X}_{m_2}\|^2}{2\sigma_2^2}\right)$$

כך ש:

$$P_{m_1, m_2} = \frac{P_{m_1|m_2} + P_{m_2|m_1}}{2N}$$

b. בחירת פונקציית הדמיון במקומות פונקציית t :

באלגוריתם שתואר, פונקציית הדמיון בין שתי נקודות במרחב U - Y נתונה על ידי התפלגות t . ניתן להגדיר גם פונקציה אחרת, למשל את פונקציית דמיון גאואיאנית עבורה שתי מידות במרחב U . שיטה זו נקראת SNE, והגדיאנטו של פונקציית המחיר במקורה זה נתונה על ידי:

$$\frac{\delta C}{\delta \vec{Y}_{m_i}} = 4 \sum_{m_1} (P_{m_1, m_i} - Q_{m_1, m_i}) (\vec{Y}_{m_1} - \vec{Y}_{m_i})$$

אולם, פונקציית דמיון גאואיאנית במרחב U יכולה לגרום לכך שנקודות לא מאד קרובות במרחב X , ימוו לנקודות קרובות במרחב U . זה קורה מכיוון שההתפלגות גאואיאנית במרחב U גורמת לאטרקטורו (משיכה) יחסי בין שתי נקודות, גם במקרים בהם הנקודות אין מודען קרובות. לעומת זאת, כאשר פונקציית הדמיון הינה התפלגות סטודנט t , שינוי התפלגות עם צוב כדי יותר, שתי נקודות שאין מודען מושפעות ימוו בצוירה ראייה למרחב U כך שאין "גמשכות" או מתקרכבות זו לזו. שיטה אחרת, הקרויה t-SNE, מציעה להשתמש בהתפלגות איחידה, אך גם להחרון דומה ל-SNE, כאשר שתי נקודות לא מאד דומות זו לזו, אין "דוחות"ichert את השניה.

לשיטת t-SNE יש שלוש מגבלות עיקריות

א. הורדת ממד: השיטה משתמש לויזואלייזציה של מידע מממד גבוה בדו-ממד או תלת-ממד. אולם, באופן עירוני, ניתן לנרצה להוריד את הממד לא לשם הצגתן, אלא לצרכים אחרים, כאשר הממד החדש הינו גדול מ-3. ניתן ובממד גבוה פונקציית התפלגות סטודנט t עם דרגת חופש אחת, אשר לה משקל גבוה יחסית

- במרחקים גבוהים, לא תשמור את המבנה של המידע המקורי. לכן, כאשר נרצה להוריד לממד גובה מ-3, פונקציית התפלגות t עם יותר מדרגת חופש אחת מתאימות יותר.
- ב. קליטת הממדיות: SNE-t מבוססת על מאפיינים קווויים בין נקודות. השיטה, המבוססת על מטריקת מרחק אוקלידי, וכך מינחה לינאריות מקומית על ידי היריעה המתמטית בה מתקיות הנקודות. אולם, במרחב נתונים בו הממד הפנימי גבוה, שיטת SNE-t עלולה ליכישל כיוון שהנחה הלינאריות לא מתקינה. לשורת שישן מספר שיטות למצוור תופעה זו, עדיין, בהגדלה, כאשר הממד הפנימי גבוה, לא ניתן להוריד ממד כך שמבנה המידע ישמר באופן מלא.
- ג. פונקציית מחיר לא קמורה: הרבה שיטות למידה מבוססות על פונקציית הפסד קמורה, כך שתיאורטית מציאת אופטימיזציה (יחידה) לפונקציה זו אפשרית תמיד. אולם, בשיטת SNE-t, פונקציית המחיר אינה קמורה, והפתרון המתkeletal על ידי האופטימיזציה משתמש בהתאם לפרמטרים הבחרם.

2.3.3 Uniform Manifold Approximation (UMAP)

UMAP הינה שיטה לא לינארית נוספת עבור הורדת ממד, המבוססת על קירוב ייירה טופולוגית (manifold). השיטה מבוססת על מספר הנחות בסיסי: ראשית, מינחים כי הנקודות/הדוגמאות מתפלגות באופן אחיד על ידי ייירה טופולוגית כלשהיא. בנוסף, מינחים כי ייירה זו קשירה באופן מקומי. מטרת קירוב זה היא לשמר את המבנה של הדוגמאות על גבי היריעה. נעיר כי הוכחה ריגורוזית של השיטה מעורבת מתמטיקה متقدמת (מעבר ללימוד בתואר הראשון), אך ניתן להבין את הרעיון העיקרי גם ללא ההוכחה המדעית, ועל כן נציג רשאית את הרעיון העיקרי, ולאחר מכן הסבירים בסיסי המתמטי העומד מאחוריו. השיטה מוחללת לשני שלבים עיקריים – בשלב הראשון מחשבים משקל של קשת בין כל שתי דוגמאות במרחב, ולאחר מכן מבצעים את הורדת הממד על מנת משקל הקשרות:

1. ראשית, נרצה לתאר את היריעה הטופולוגית בה נמצאות כל הדוגמאות במרחב המקורי, \hat{X} , על ידי גראף ממושקל-קיים. על מנת ליצור גראף זה יש לבצע את השלבים הבאים עבור כל דוגמא X_{m_i} (שהיא כאמור נקודה במרחב המקורי- N -ממדי):

- א. מציאת את k השכנים הקרובים ביותר על ידי שימוש במרחב האוקלידי (\mathbb{R}^d):
 $d_{i,j} \equiv d(X_{m_i}, X_{m_j})$
 ב. חישוב המרחק לנקודה הקרובה ביותר – נסמן אותו ב- r .
 ג. חישוב הגודל σ_i על ידי פירעון המשוואה:

$$\log_2(k) = \sum_{j=1}^k \exp\left(-\frac{\max\{0, d_{ij} - \rho_i\}}{\sigma_i}\right)$$

המשקל (הסימטרי) של קשת בין שתי נקודות – X_{m_i} ו- X_{m_j} – :

$$P_{ij} \equiv P_{i|j} + F_{j|i} - P_{i|j} \cdot F_{j|i}$$

כאשר:

$$P_{i|j} = \exp\left(-\frac{\max\{0, d_{ij} - \rho_i\}}{\sigma_i}\right)$$

כלומר: עבור X_{m_j} שאינו בקבוצת k השכנים הקרובים של X_{m_i} מתקיים: $P_{i|j} = 0$, ואם שתיהן בנקודות הן כן "שכנות", אז מתקיים: $P_{i|j} = \exp\left(\frac{\rho_i - d_{ij}}{\sigma_i}\right)$

2. כעת, על מנת לבצע הורדת ממד, נרצה לייצר הצגה של הגראף הממושקל בממד נמוך מרחב \hat{Y} , כאשר לגרף החדש יש "מבנה דומה" לגרף המקורי. לשם כך, נגדיר פונקציית דמיון באופן הבא:

$$q_{i,j} \equiv \left(1 + a \|\vec{Y}_{m_1} - \vec{Y}_{m_2}\|^{2b}\right)^{-1}$$

כאשר פונקציית המחיר הינה:

$$C = \sum_{i \neq j} P_{ij} \log\left(\frac{P_{ij}}{q_{ij}}\right) + (1 - P_{ij}) \log\left(\frac{1 - P_{ij}}{1 - q_{ij}}\right)$$

עיר כי מספר השכנים הקרובים k , והמרחב המינימלי האפקטיבי (הנקבע על ידי a, b) הינם היפר-פרמטרים הנקבעים על ידי המשתמש.

הבדלים העיקריים בין שיטת SNE-t לUMBAP:

- א. בשיטת SNE-t המרחקים האוקלידיים מחושבים כל זוג נקודות. בשיטת UMAP, נלקחים בחשבון רק קבוצת שכנים הקרובים.
- ב. הסימטריזציה בשיטת SNE-t נתונה על ידי: $P_{i,j} = \frac{P_{i,j} + P_{j,i}}{2N}$. לעומת זאת, הסימטריזציה בשיטת UMAP נתונה על ידי $P_{i,j} \equiv P_{i,j} + F_{j|i} - P_{i|j} \cdot F_{j|i}$.
- ג. פונקציית הדמיון במרחב SNE-t הינה גאומטרית. לעומת זאת, בשיטת UMAP פונקציית הדמיון במרחב \mathcal{X} הינה אקספוננטית.
- ד. עבור UMAP פונקציית המשקל (הדמיון) במרחב החדש \mathcal{Y} הינה $q_{i,j} \equiv \left(1 + a \left\| \vec{Y}_{m_i} - \vec{Y}_{m_j} \right\|^{2b}\right)^{-1}$ כאשר $a = b$ ונוסף פונקציה זו מormalת (מוכפלת במקדם -3), אנו מקלימים את פונקציית המשקל של מרחב \mathcal{Y} עבור SNE-t.
- ה. לפונקציית המחריר (או הפוטנציאלי) עבור שיטת UMAP נוסף איבר מהצורה $(1 - P_{i,j}) \log \left(\frac{1 - P_{i,j}}{1 - q_{i,j}} \right)$. איבר זה מוריד את סיכוי לקבל שתי נקודות קרובות מאוד במרחב החדש (זאת אומרת כאשר $1 - q_{i,j}$ אינן קרובות במרחב המקורי).

תיאור מתמטי של השיטה:

נניח שהיריעה הטופולוגית בה המידע (data) נמצא, אינה ידועה מראש, אולם בכל זאת נרצה להגדיר מרחקים גאומטריים על גבי יריעת זו. בהתאם, נניח כי הדוגמאות מפוזרות באופן אחד על גבי ירעה זו. באופן כללי, הגדרת מטריקת רימן (מטריקה, פונקציה מסויימת, שמאפשרת לנו הנדרשת מרחוקם, ראה דוגמא למטה), תליה בסביבת הנקודה. המטרה הראשונית שלנו היא להגדיר את היריעה הטופולוגית בה הנקודות נמצאות על ידי גוף ממושך קשיי.

כעת, נניח שבבסיסבת נקודה x המטריקה g_x אלכסונית וקבועה. לכן, ניתן להגדיר את המרחק בין נקודה x לנקודה q שמנמצאת בסביבת x ע"י המטריקה הקיימת במרחב ה- M מדדי:

מתוך ההנחה שהנקודות מפוזרות בצורה אחידה על גבי היריעה, מתקיים הדרישה הבאה. רדיוס של כדור סביב נקודה x (הדוגמה ה- i -ית של מערכ הנתונים הרב מדדי) כך שהכדור מכיל k שכנים קרובים ל- x , הוא קבוע ואינו משתנה בין הנקודות.

אם כך, בידונו משפחת מטריקות מקומיות, אחת לכל דוגמא i -ית, שעל ידה ניתן להגדיר את המרחקים ל- k שכנים קרובים. רצוח "לאחד" את משפחת המטריקות המקומיות הללו, למרחב גלובלי. לשם כך, נשתמש ב-simplicial sets [לצורך הדיון קבוצות אלו מכילות סימפלקס – מבנה מסוים של גראף/רשת – כך שלכל איבר יש דרגת שייכות לקבוצה]. נשים לב כי עלי ידי איחוד הקבוצות הללו, צרמו גוף ממושך-קשיר. עבור גוף זה, כל נקודה מקושרת אר ורך לקבוצת k השכנים הקרובים לה, כאשר עבור שכנים קרובים יותר, משקל הקשת המחברת גבוהה יותר ממשקל קשת עבור שכנים רחוקים יותר. נשים לב כי חישוב המשקלים ו"קישור" עבור שכנים קרובים בלבד מאפשרת לשחזר את מבנה היריעה הטופולוגית ביותר דיוק (ראה איור).

נניח, בדומה לפירק הקודם, כי מערכ הנקודות המקורי מסומן על ידי, $\vec{X}_M \in \mathbb{R}^{M \times N}$, כאשר M הוא מספר הדוגמאות, ו- N הוא מספר המאפיינים (או המשתנים). חשוב לשים לב כי כל מדידה מייצגת על ידי וקטור שורה \vec{x}_m . נרצה למפותה מדידות אלו ל- $\hat{\mathcal{X}} \in \mathbb{R}^{M \times D}$ – במרחב המוקורי \mathcal{X} : $\hat{\mathcal{X}}_i = \vec{Y}_1, \dots, \vec{Y}_n$, כך שמדידה i במרחב המוקורי \mathcal{X} , מזאת על ידי וקטור חדש \hat{y}_i במרחב החדש \mathcal{Y} , כך שמתקדים $N < D$, זאת אומרת הממד של הווקטור החדש הרבה יותר קטן מהමמד המקורי.

כעת, על מנת להשוות בין ייצוג ב-fuzzy simplicial sets עבור $\hat{\mathcal{X}}$ ו- \hat{Y} , נשתמש ב-cross entropy. נניח שדרגת השתייכות לאיבר a בקבוצה A נתון על ידי פונקציה $A \rightarrow [0, 1]$: $a \mapsto \mu$. הימן המקבול עבור cross entropy על (A, μ) ו(A, ν) מוגדרת להיות:

$$C[(A, \mu), (A, \nu)] \equiv \sum_{a \in A} \left[\mu(a) \log \left(\frac{\mu(a)}{\nu(a)} \right) + (1 - \mu(a)) \log \left(\frac{1 - \mu(a)}{1 - \nu(a)} \right) \right]$$

עבור קבוצת המדידות במרחב המקורי: $\{\vec{X}_1, \dots, \vec{X}_M\}$, נסמן מטריקה $\{0\} \cup \hat{\mathcal{X}} \times \mathbb{R}^+$ המכתת את א-הדים בין שתי מדידות; קרי מדידות דומות יציגו מרחק קטן ביניהן, לעומת זאת מדידות שאינן דומות בהן המרחק המוגדר על ידי המטריקה ימצא להיות גדול. נגיד תחת-קבוצה של k השכנים הקרובים, על פי המטריקה d , של

מידה \vec{X}_i , ונסמנה באופן הבא: $\{\vec{X}_{i_1}, \vec{X}_{i_2}, \dots, \vec{X}_{i_k}\}$. עבור כל מידה \vec{X}_i , נחפש את המרחק הקטן ביותר בינו לבין השכנים שלה, זאת אומרת:

$$\rho_i = \min \left\{ d(\vec{X}_i, \vec{X}_{i_j}) : 1 \leq j \leq k, d(\vec{X}_i, \vec{X}_{i_j}) > 0 \right\}$$

בנוסף, נחפש σ_i כר שמתקיים:

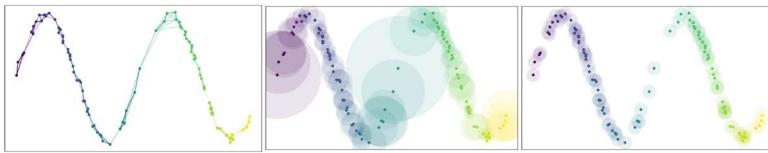
$$\sum_{j=1}^k \exp \left[-\frac{\max(0, d(\vec{X}_i, \vec{X}_{i_j}) - \rho_i)}{\sigma_i} \right] = \log_2 k$$

כעת, נוכל להגדיר גראף מכוון ומ משקל (weighted directed graph) $\bar{G} = (\bar{V}, \bar{E}, \bar{w})$ שקבוצת הנקודות V הינה קבוצת המדידות \hat{X} , קבוצת הקשרות (edges) נקבעת על ידי k השכנים הקרובים לכל נקודת מדידה:

$$E = \{(\vec{X}_i, \vec{X}_{i_j}) : 1 \leq i \leq M, 1 \leq j \leq k\}$$

והמשקל עבור כל קשת מוגדר על ידי:

$$w[(\vec{X}_i, \vec{X}_{i_j})] = \sum_{j=1}^k \exp \left[-\frac{\max(0, d(\vec{X}_i, \vec{X}_{i_j}) - \rho_i)}{\sigma_i} \right] \equiv p_{i|j}$$



איור 2.16 לפניכם נקודות הנמצאות על גבי יריעה טופולוגית שנייה בקשר בקשרות סינפס. אייר מיימן ייתן לנווט כי הנקודות מפוזרות באופן אחיד על גבי יריעה זו, אך אם נקשר רק נקודות שכנות (הנמצאות בתחום סביבת נקודה) נקבל גראף / יריעה שאינה קשירה. אם לעומת זאת נדרש שכבות תקווינה מסוימת k שנים קרובים הינו כדור בעל רדיוס מסוים, נמצא כי רדיוס כדור זה אינו קבוע בין הנקודות (ראה איור אמצעי). לשם כך, הינה הבסיסית של שיטת UMAP הינה שהנקודות מפוזלות בזרה אחידה על גבי היריעה. כעת, לאחר שהנתנו הפלות אחידה של הנקודות על גבי היריעה, נוכל להגדיר את משקל הקשות בין שניים קרובים (אהר שמאל). נשים לב כי באופן היריעה המתואר עליי הגרף הקשור אכן נראה במצב סינפס כדרש.

נספח מתמטי:

בහינת מטריקת רימן (x, y) γ עבור סביבת נקודה x על גבי היריעה. האורן של העוקום (x, y) נתנו ע"י:

$$\|\gamma(x_i, x_f)\|_g \equiv \int_{x_i}^{x_f} \sqrt{\sum_{\mu} \sum_{\nu} g_{\mu, \nu}(p) dx_{\mu} dx_{\nu}} = \int_{x_i}^{x_f} \sqrt{\sum_{\mu} \sum_{\nu} g_{\mu, \nu}(p) \frac{\partial x_{\mu}}{\partial t} \frac{\partial x_{\nu}}{\partial t}} dt$$

כאשר הסכימה הכפולה (v, μ) על קווטורי הבסיס של העוקמה. נבחר את העוקמה שנותנת מרחק מינימלי.

דוגמאות:

דוגמא ליריעה דו-ממדית הינה משטח אוקלידי דו-ממד \mathbb{R}^2 . מטריקת רימן עבור ירעה זו, אינו תלוי בנקודת x , ומוגדר על ידי: $g_{\mu, \nu} = \delta_{\mu, \nu}$. עבור עוקום במישור הדו-ממד: $(x, y) = (x, f(t))$. פרמטריזציה של עוקום זה נתונה על ידי:

אור עוקום זה נתון על ידי:

$$\|\gamma(x_i, x_f)\| \equiv \int_{x_i}^{x_f} \sqrt{\sum_{\mu=x,y} \sum_{\nu=x,y} \delta_{x,y} \frac{\partial x_{\mu}}{\partial t} \frac{\partial x_{\nu}}{\partial t}} dt = \int_{x_i}^{x_f} \sqrt{\left(\frac{\partial x}{\partial t}\right)^2 + \left(\frac{\partial f(t)}{\partial t}\right)^2} dt$$

עקומה (t) γ שנותנת מרחק מינימלי בין נקודות מוגדרת על ידי (n, m) , כך:

$$\int_{x_i}^{x_f} \sqrt{1 + m^2} dt = \sqrt{1 + m^2} (x_f - x_i) = \sqrt{(x_f - x_i)^2 + m^2 (x_f - x_i)^2} = \sqrt{(x_f - x_i)^2 + (y_f - y_i)^2}$$

כמפורט.

גם ספירה ד-ממדית (מוסמנת לעיטים על ידי \mathbb{S}^2 , לדוגמה גלובוס) מהויה ד- ממדית. שם, מטריקת רימן מוגדרת על ידי
לכ', עבר עקום על גב גלובוס; $(t) = (\phi, \theta)$ נקבע:

$$\|\gamma\| \equiv \int_{x_i}^{x_f} \sqrt{\left(\frac{\partial \theta}{\partial t}\right)^2 + \sin^2 \theta \left(\frac{\partial \phi}{\partial t}\right)^2} dt$$

2.4 Ensemble Learning

2.4.1 Introduction to Ensemble Learning

נניח ויש בידינו אוסף נתונים מסוים, וחיצים לבנות מודל המנתה את הנתונים האלו שמתבסס על אלגוריתם מסוים. כמעט תמיד, המודל לא יהיה מדויק במידה כלשהי, והוא יהיה בעל שונות או בעל הטיה. ניתן להשתמש במקרה (Ensemble) של מודלים שונים המבוססים על אותו אלגוריתם רצוי, ובכך לקבל מודל משוקל בעל שונות/טיה נמוכים יותר מאשר מודל שמתבסס על אותו אלגוריתם אך בנבנה באופן פשוט.

בדי' להבין את יותר טוב את החשיבות של שימוש ב-ensembles, יש להרחיב על-hoof Trade-off בין שונות המודל להטיה שבו. מודל אופטימלי יתאפשר בשונות נמוכה ובטיה גבוהה. לעומת זאת, השוני בין התוצאות לא יהיה מהותי, ובמוצע ההחיזיות תהיה קרובת מזו של הטעינה האמיתית. מודל כזה יהיה מודל אמין, וכן ככל שיהיה יותר צעדים. למרבה הצער, מודל שזכה לרובה אין אפשרי. סוג אחר של מודל יהיה המודל הגראונט, ההפוך למודל אוטומטי. זהו מודל עם שונות גבוהה והטיה גדולה. מודל שכזה יציג טווח רחב של תוצאות על אותם נתונים, ובמוצעו יהיה הרחק מאוד מהערך האמיתי. מודל זה כל אינו שימושי.

בפועל, המודלים במציאות ינושו לאורך שני קצווים: מודלים עם שונות נמוכה, ומודלים עם שונות גבוהה והטיה גבוהה. הזרוי של המיקום שלם לאורך ציר זה קרייטי, כיון שהוא מאפשר לנו לבחור את דרך ההסתמודות הטובה ביותר. שמספר משפחות של models ensembles, ושני העקריים שבהם נקראים "מודלים Bagging and Boosting". כאשר ניתן במודלים עם שונות גבוהה, ככלומר מודל הסובב-overfitting, לרוב מודל מושג-m-Boosting, לרוב מודל המשמש באנSEMBLE בענוגת Bagging. אלגוריתם מסוג Boosting טיפול במרקחה השב, בו הטעינה גבוהה והשונות נמוכה.

2.4.2 Bootstrap aggregating (Bagging)

Bagging היא משפחת אלגוריתמים אשר פועלת כ-ensemble, כלומר – מספר אלגוריתמים שפועלים ביחד, על-מנת להיעיל לתוצאה מסוימת. כאמור, אלגוריתמים מסוג bagging נועד להגדיל את יציבות המודל והעלאת הדיווק שלו, זאת תוך הורדת השונות והימנענות מ-overfitting. Bagging מושך מספר רב של אלגוריתמים, המכונים "מודלים חלשים" (Weak learners), כאשר כל אחד מהם מבצע למידה ותחזית על חלק מן הנתונים, מתרז מטריה להציג תוצאה אובייקטיבית. אלגוריתם bagging הינו שיטה נפוצה ופשוטה יחסית לשיפור ביצועים, אם כי הוא עשוי להיות יקרר בבחינה חשיבתית.

מודל "פשוט" יוכל את הנתונים, יתאמן עליהם ויבצע תחזית על נתונים חדשים. זהו תהליך הלמידה וה מבחנן אשר ידוע לנו מודלים כגון עץ החלטה (Decision Tree), גראטיה לינארית וכו'. כפי שהסביר לעיל, מצב כזה עשוי להוביל להסתמאות יתר של המודל נתונים האימון, דבר שעשוי להשפיע על שונות גבוהה. בכך להתמודד עם בעיה זו, אלגוריתמים מסוג bagging ומתקנים את התהליך לשני שלבים: Bootstrapping and Aggregating.

בשלב ה-bootstrapping, יוצרם מהנתונים המקוריים קבוצות חדשות, כאשר כל קבוצה נוצרת על ידי דוגימה (עם חזירות) של איברים מהקבוצה המקורית, באופן כזה שגודל כל קבוצה חדשה הוא בגודל של הדadataה המקורית. בשלב השני, Aggregating-the, הקבוצות החדשנות ננסחות כקלט ל"לומנים חלשים", אלגוריתמים פשוטים יותר, אשר עובדים במקביל על תחזית, ככלומר, יוצרים מודל נפרד לכל קבוצה של נתונים. בשלב הסופי, יבצע איחוד של כל המודלים על מנת ליצור מודל משוקל בעל שונות קטנה יותר מאשר מודל המסתמך על הדadataה המקורית כפי שהוא.

אוף חיבור המודלים המתבצע בbagging מבוסס על אותו רעיון של NN-K, כאשר מודלים אלו יוכולים לשמש הן למטרות סיוג וכן למטרות רגסיה. כאמור לעיל (פרק 2.1.3), באלגוריתם השן הקרוב לכ"סן" העד לעתותיו שלו, ולאחר הרכעת הרוב ובבעה התוויות של התוצאות החדשנה. בברקירה שבו ונפairo את דדיות כל התוויות השונות, התוויות הנבחרת תהיה של התוצאות הנouceה ביותר; נשעה זאת כאשר NN-K-NN. יעבד כמסוג. במקרים בהם NN-K-NN יעבד כרגסיה, יבצע ממוצע של כל התוויות השונות, וזאת גם תהיה התחזית. כאשר יעבד כמסוג, כל

bagging weak learner יבצע תחזית, והתוויות השכיחות ביותר תהיה התוצאה של האנסמבל (-הכרעת הרוב). כאשר learner יעבדו כרגסיה, כל מודל יבצע תחזית, אבל התוצאה של האנסמבל תהיה הממוצע של כל המודלים.

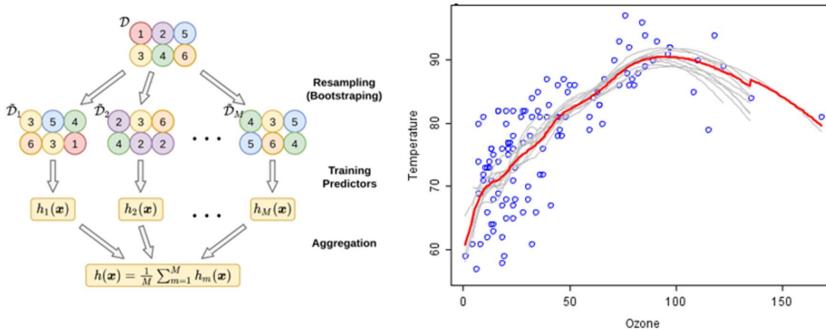
באופן פורמלי, עבור דוגמה $\mathcal{D} \in \mathbb{R}^{n \times d}$, ניצור M קבוצות חדשות באוטו גודל של הדוגמא המקורי – עבור כל קבוצה $X_m \in \mathbb{R}^{n \times d}$, נבנה מודל $c_m(x)$. עבור עייניות סיווג החלטה התקבל על פי הצעת הרוב:

$$C(x) = \text{majority}(\{c_1(x), \dots, c_M(x)\})$$

עבור עייניות רגסיה ההחלטה תבוצע בעזרת מיצוע כל המודלים:

$$C(x) = \frac{1}{M} \sum_{m=1}^M c_m(x)$$

a



איור 2.17 (a) אלגוריתם Bagging: בשלב ראשון יוצרים הרבה מחלקות שונות שנותרונות מהדוגמא המקורי (Bootstrapping), ולאחר מכן מודל המתאים לכל מחלקה אחד (Aggregating), ולבסוף יוצרים מודל יחיד המבוסס על כל המודלים המקוריים. (b) דוגמא לבניית מודל רגסיה בעזרת אלגוריתם Bagging. ניתן לראות שהמודל המשקלן הוא בעל שונות קטנה יותר מאשר מודלים.

בין אם משתמשים בהכרעת הרוב ובין אם משתמשים בmixing של המודלים, המודל המשקלן שנוצר יהיה חלק יותר ובול פחות שיפורים חדים, מה שמקטין את ה-overfitting, ומミיאן מפחית את השונות. ניתן להסביר זאת על ידי דוגמא פשוטה – נניח שיש התפלגות נורמלית (σ^2, μ). נזק השונות של A דגימות בלבד תלויה הינה $\frac{\sigma^2}{n}$. השונות הממוצעת נניח ובמצאים m ניסויים שככל מהם דוגמים A נקיים, ובין כל שתי קבוצות יש קוורציה ρ . השונות הממוצעת של הניסויים הינה:

$$\text{Var}\left(\frac{1}{m} \sum_{i=1}^m \text{single cycle}\right) = \frac{1}{m}(1 - \rho)\sigma^2 + \rho\sigma^2$$

אם נבצע הרובה מאד ניסויים, ככלומר ניקח m גדול מאוד, נקבל:

$$\lim_{m \rightarrow \infty} \frac{1}{m}(1 - \rho)\sigma^2 + \rho\sigma^2 = \rho\sigma^2$$

ובכן הכל השונות הסופית הינה בקירוב $\rho\sigma^2$, וביתוי זה לרוב קטן מאשר השונות של מודל המבוסס על הדוגמא המקורי לא לשימוש ב-ensembles. ניתן לשמש לב שכך שהקוורציה בין הקבוצות קטנה, כך השונות של המודל המשקלן גם כן קטנה יותר.

מודול וגוף מאוד מסווג bagging הוא Random Forest. אלגוריתם זה משלב בין עצי החלטה לבין הרעיון הבסיסי של bagging, כאשר הוא מפצל את הנתונים ואת המשתנים לעצי החלטה רבים, וכל אחד מהם מקבל חלק מסוים מן השלים. העצים הם בעלי שונות גבוהה, ככלומר – כל אחד מהם הוא overfitting עצמו, אך עם זאת הקוורציה ביניהם נמוכה, מה שמקל על הורדת השונות והימנע מ-overfitting. לבסוף, השקלול של כל המודלים ביחס מצילח לייצר מודל בעל שונות נמוכה, וומוביל לתוצאות טובות.

במקרה אחד, על מנת לאפשר לנו לחשוף נתונים לא-ordinaries (Outliers), ניתן להשתמש בbagging. הוא מושך את המינימום השוני, והוא גם מסייע לאריכים קיצוניים (Outliers).

תינן עדיפות למודלים פשוטים יותר של ensembles. להיות יקר מבחינה חישובית. עקב לכך, הוא שימושו מאוד במקרים בהם שיפור צער עשוי להיות לחייב הצלחה, אך לרוב יישר את ההתייה, וכן עשוי לא להתאים במקרים רבים. מודלים של *bagging* יישם גם במקרה זה.

2.4.3 Boosting

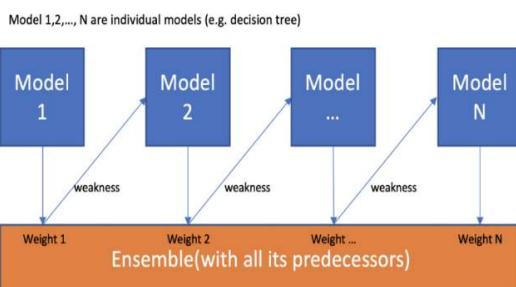
כל לומד חלש לומד חזק על ידי בניות קומבינציה ליבורטיא של מוסוגים אשר נוצרו בעזרת הלומד החלש.

נכחיש את הרעיון של boosting בעזרת דוגמא: נניח שיש בידינו אוסף נתונים X , המוחלוק באופן אקראי לשולש קבוצות (כל אחת מכילה שלוש מהנתונים) – x_1, x_2, x_3 .icut בונים מודל ליצור סיווג ביןאירי, המסמן ב- h_1 . נמצא כי x_1 מותאים בaczורה טובה רק לקבוצות x_2, x_3 אך מסוכן בaczורה לא טובה את פרטיה הקבוצתי x_3 . כיוון ש- x_3 מכיל שלוש מסק הנתונים, שגיאת הסיווג גדולה ו-(x_1) הוא מודל חלש, ונרצה לשפר אותו. בצד לעשות זאת ניקח רק חלק מהנתונים, X' , ונdag לאיך ש- X' יכול הרבה מאיברי x_3 .icut בונה מודל נוסף (x_2 על בסיס X' , מתקו כוננה שמודל זה יימחק גם בקבוצת x_3 בסיווג את אירוחה בaczורה טובה.icut בנתנו now שמודל זה אכן מסוכן בaczורה ואוותה את איברי x_3 , אך הפעם המודל שוגה בaczורה גסה בסיווג איברי x_2 .uck השגיאה בסיווג x_2 הא מודל העשוי גם הוא מודל חלש, אךicut יש בידינו שני מודלים חלשים שהחוולשה בכל אחד nowות מקבוצת איברים אחד של אוסף הנתונים המהווק X . אם נמצא דרך הולמת לחסר את שבי המסתויים, ככל ליצור מודל בעל פוטנציאל להציגי הסיווג את X כמו שאנו בראנו.

באופן כללי, אם רוצים לאמן מודל $(\mathcal{X}, \mathcal{C})$ באמצעות אלגוריתם L על אוסף הנתונים \mathcal{D} , יש לבצע את השלבים הבאים:

- .1. אתחול הנתונים: $\mathcal{D}_1 = \mathcal{D}$
 - .2. עברו $t = 1, \dots, T$
 - .3. אימון מודל חישוב על $c_t(x) = L(\mathcal{D}_t)$
 - .4. חישוב שגיאת המסואג: $\epsilon_t = P_{x \sim \mathcal{D}_t}(c_t(x) \neq f(x))$
 - .5. התחמת הנתונים עבור האיטרציה הבאה: $\mathcal{D}_{t+1} = \text{Adjust Distribution}(\mathcal{D}_t, \epsilon_t)$
 - .6. איחוד המודלים החלשים: $\mathcal{L}(x) = \text{combine outputs}\{c_1(x), \dots, c_T(x)\}$

יש כל מיני שיטות כיצד לבצע את השלבים השונים באלגוריתם boosting, ונפרט את המרכזיות שבהן.



איור 2.18 – סכמת כללית של boosting. המודלים (במקרה זה מדובר בעץ החלטה רדו-ודו, אך זה תקף לכל מודל חלש) מחוברים אחד לשני באופן שכל אחד לומד מהתפלגות המשקלות בהתאם לשגיאות של המודלים הקודמים.

Adaptive-Boosting (AdaBoost)

AdaBoost היא אחת הטכניקות הראשונות של boosting, ועל אף^K שקיימות טכניקות נוספות (במיוחד ביחס למודלים מושגים) היא בין הפופולריות ביותר בתחום (אם כי יש לה מספר מוגבלים של מודלים). העצמה הגלומה בטכניקה זו נובעת מכך שגם בהינתן מספר מאפיינים רב, האלגוריתם מצליח להשיג פחوت מ"קלות המודדים" ולשמור על יכולות ייבוא טובות, ב向きו לאלגוריתמים אחרים של סיווג, כמו למשל SVM או אפילו רשתות נeurונים.

בדoor, תחת ההנחה שקיים אלגוריתם לומד חלש (x, c), המטריה היא למצוא דרך להפוך אותו למודל חזק (\mathcal{C}). באופן אנטואטיבי היה ניתן לחשב שאפשר מודלים על תת קבוצות של הדאטा המקרים (עם אפשרות לוחיפויות בין תת-קבוצות), להשתמש ב-vote-majority, ובכך לשרשר את ההיפותזות של כל המודלים לפטל אחד. גישה זו מבון אינטואטיבית ופסנתרית, ואינה לוקחת בחשבון מרבית המודלים השונים. גישה טובה יותר תהיה לבנות מודל על בסיס חלק מהדאטא, לבחון את מידת הצלחה של המודל על יתר הדאטא, ולפי ההצלחה שלו במשימה זו לחתם משקל ל-vote של המודל. ניתן להסביר תחוכם לרועין זה, כך שbulk של ביבינון יותר דגש איברים בדאטא שהמודלים הקיימים יושבו בסיווג שלהם, ובכך בכל שלב בו מאמנים מודל נוסף תורח הצלחה יותר גוזלה. AdaBoost מאשר המודל הקודם. חלק זה הינו החלק האדפטיבי (Adaptive) באלגוריתם, על שמו נקרא האלגוריתםAdaBoost.

כעת, נסביר כיצד ניתן להרכיב מסווג חזק באמצעות אוסף של מסווגים חלשים עבור אוסף נתונים $\mathbb{R}^N \in X$.

1. ראשית יש לאותחל משקלות באופן אחיד עבור כל אחד מ- N הדוגמאות בסט הנתונים $w_i^{t=0} = \frac{1}{N}$.

2. לאחר מכן יש לבצע איטרציות באופן הבא:

בכיתה מסווג אופטימלי (c_t) ביחס לאוסף הנתונים המשקלל.

чисוב שגיאת הסיווג של ($c_t(x) \neq y_i$):

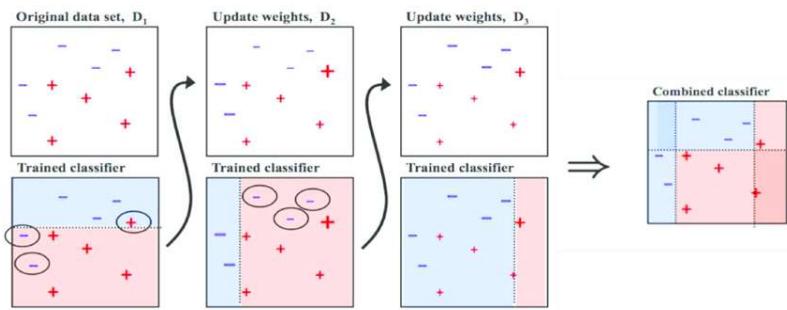
$$\alpha_t = \frac{1}{2} \ln \left(\frac{1-\epsilon}{\epsilon} \right)$$

עדכון המשקלים: $w_i^{t+1} = w_i^t \exp(-\alpha_t y_i c_t(x))$

נרטול המשקלים בהתאם לסכום הכלול: $N_{t+1} = \sum_i w_i^t \rightarrow w_i^{t+1} = \frac{w_i^t}{N_{t+1}}$

3. חישוב המסווג המשקלל, שהוא קומבינציה ליארית של החומרוגיה החלשים:

$$C(x) = \text{sign} \left(\sum_t \alpha_t c_t(x) \right)$$



איור 2.19 – דוגמא לשימוש בעבור מודל סיווג ביארי.

2. References

SVM:

https://commons.wikimedia.org/wiki/File:Svm_max_sep_hyperplane_with_margin.png

<https://svm.michalhaltuf.cz/support-vector-machines/>

<https://medium.com/analytics-vidhya/how-to-classify-non-linear-data-to-linear-data-bb2df1a6b781>

https://xavierbourretsicotte.github.io/Kernel_feature_map.html

Naïve Bayes:

https://en.wikipedia.org/wiki/Naive_Bayes_classifier

https://scikit-learn.org/stable/modules/naive_bayes.html

K-NN:

<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>

EM:

https://www.cs.toronto.edu/~urtasun/courses/CSC411_Fall16/13_mog.pdf

https://stephens999.github.io/fiveMinuteStats/intro_to_em.html

Hierarchical Clustering:

<https://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/>

LOF:

<https://towardsdatascience.com/local-outlier-factor-lof-algorithm-for-outlier-identification-8efb887d9843>

PCA:

Saal, L.H. et al. (2007). *Proc. Natl. Acad. Sci. USA* 104, 7564–7569.

3. Linear Neural Networks

פרק זה עוסק בבעיות רגסיה – כיצד ניתן בעזרת אוסף דוגמאות נתון לבנות מודל המסוגל לספק מידע על נקודות חדשות שיגעו ויחסר עליהם מידע. המודלים שייצגו בפרק זה מתייחסים למצוא עבורו הפרדה לינארית, כלומר, ניתן למצוא קווים לינאריים המחלקים את הדatta לשתי קבוצות שונות. החלק הראשון של הפרק עוסק ברגסיה לינארית (Linear regression) והחלק השני עוסק ברגסיה לוגיסטי (Logistic regression). לבסוף יציג מבנה שקול לביעות הרוגטיה בזרת רשת ניירונים פשוטה, ובונה זה יהיה הבסיס לפיק האוסף ברשומות נירונים עמוקות, הבאות להתמודד עם דatta שאינו ניתן לבצע עבורו הפרדה לינארית.

3.1 Linear Regression

3.1.1 The Basic Concept

המודל הפשוט ביותר הינו linear regression. מודל זה מנסה למצוא קשר לינאריבן במספר משתנים או מספר מאפיינים. בהנחה שמתיקים יחס לינאריבן סט משתנים בלתי תלויים $\mathbb{R}^d \in x$, ניתן לכתוב את הקשר ביןיהם כזורה הבאה:

$$\hat{y} = w^T x + b = w_1 x_1 + w_2 x_2 + \dots + w_d x_d + b$$

כאשר $w \in \mathbb{R}^d$ הם המשקלים ו- $b \in \mathbb{R}$ נקרא bias

דוגמא: ניתן לטעון כי מחיר הבתים באזרע מסוים נמצא לינאריבן במספר פרמטרים: גודל הדירה,อายוזה קומה היא נמצאת, וכמה שנים הבניין קיימ. תחת הנחה זו, שלח奸 את המודול עבור דוגמאות ידועות ובכך למצוא את המשקלים וה-bias. לאחר מכן ניתן לקחת את המודול ולנחש את מחיר הדירה עבור נתונים שונים לא ידוע, אך הפרמטרים שלהם כן נתונים.

בדי' לבנות מודל המאפשר לשערר בצורה טובה את y בהינתן סט מאפיינים, יש לדעת את המשקלים וה-bias. כיוון שהם לא ידועים, יש לחשב אותן בעזרת אוסף של דוגמאות ידועות. ראשית יש להגדיר פונקציה מחיר ($Loss$), הנקובעת עד כמה הביצועים של מודול מסוים טובים. פונקציית המחיר היא פונקציה של הפרמטרים הנמלדים - $L(w, b)$, והבאתה למינימום טפק את הערכות האופטימליים של המשקלים וה-bias. פונקציית מחיר מקובלת הינה השגיאה הריבועית הממוצעת (MSE) – המכשנת את ריבוע ההפרש בין החיזוי לבין הפלט האמתי:

$$L^{(i)}(w, b) = \frac{1}{2}(y_i - \hat{y}_i)^2$$

כאשר נתונות n דוגמאות ידועות, יש לסכום את כל ההפרשים הללו:

$$L(w, b) = \frac{1}{n} \sum_{i=1}^n \frac{1}{2}(y_i - \hat{y}_i)^2 = \frac{1}{n} \sum_{i=1}^n \frac{1}{2}(y_i - w^T x_i - b)^2$$

כעת בשבייל למצוא את הפרמטרים האופטימליים, יש למצוא את w , b שambilאים את פונקציית המחיר למינימום:

$$\hat{w}, \hat{b} \equiv \hat{\theta} = \arg \min L(w, b)$$

עבור המקרה הסקלרי בו $d = 1$, כלומר יש מאפיין אחד ומונחים למצוא קשר בין פלט מסוים, הקשר הלינאריארי הוא $y = wx + b$. עבור המקרה זה, פונקציית המחיר תהיה:

$$L(\theta) = \frac{1}{n} \sum_{i=1}^n \frac{1}{2}(y_i - w^T x_i - b)^2$$

ובדי' למצוא אופטימום יש לגזר ולהשווות ל-0:

$$\frac{\partial L}{\partial w} = \frac{1}{n} \sum_{i=1}^n (y_i - w^T x_i - b) \cdot (-x_i) = 0$$

$$\frac{\partial L}{\partial b} = \frac{1}{n} \sum_{i=1}^n (y_i - w^T x_i - b) \cdot (-1) = 0$$

מתகבלות סט מושוואות לינאריות:

$$\begin{aligned} w \sum x_i^2 + b \sum x_i &= \sum y_i x_i \\ w \sum x_i + bn &= \sum y_i \end{aligned}$$

ובכתיב מטריציוני:

$$\begin{pmatrix} \sum x_i^2 & \sum x_i \\ \sum x_i & n \end{pmatrix} \begin{pmatrix} w \\ b \end{pmatrix} = \begin{pmatrix} \sum y_i x_i \\ \sum y_i \end{pmatrix}$$

על ידי הצבה של הדוגמאות הנתונות ניתן לקבל את הפרמטרים של הקשר הלינארי.

לשם הנוחות ניתן לסמן את ה-bias כפרמטר נוספת:

$$\hat{y} = w^T x + b = (w^T b) \begin{pmatrix} x \\ 1 \end{pmatrix} = \tilde{w}^T \tilde{x}, \quad \tilde{w}, \tilde{x} \in \mathbb{R}^{d+1}$$

עבור המקרה הווקטור ישנו דוגמאות, כלומר ש- x מאפיינים בלתי תלויים ומנסימים למצוא את הקשר ביןיהם לפחות מסוים. במקרה זה $(x_1, \dots, x_n)^T, Y = (y_1, \dots, y_n)^T$, ופונקציית המבחן הינה:

$$L(\theta) = \frac{1}{n} \sum_{i=1}^n \frac{1}{2} (y_i - w^T x_i)^2$$

המינימום של הביטוי הזה שקול למינימום של $\|Y - Xw\|^2$:

$$\frac{\partial L}{\partial w} = \frac{1}{n} \sum_{i=1}^n (y_i - w^T x_i) \cdot (-x_i) = 0$$

$$\rightarrow X^T(Xw - Y) = 0$$

$$\hat{w} = (X^T X)^{-1} X^T Y$$

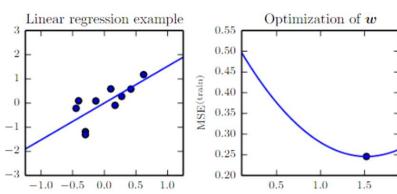
ובהינתן אוסף דוגמאות:

$$X = \begin{pmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix}, \quad \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

אזי הפתרון של הרגרסיה הלינארית הינו:

$$\hat{w} = (X^T X)^{-1} X^T Y = \left(\begin{array}{cc} \sum x_i^2 & \sum x_i \\ \sum x_i & n \end{array} \right)^{-1} \left(\begin{array}{c} \sum y_i x_i \\ \sum y_i \end{array} \right)$$

דוגמא למציאת קו הרגרסיה והמשקל האופטימלי עבור בעיה סקלרית:



איור 3.1 רגרסיה לינארית אופטימלית עבור אוסף דוגמאות נתון (שמאל) ואופטימיזציה עבור המשקל w ביחס לפונקציית המבחן (ימין).

3.1.2 Gradient Descent

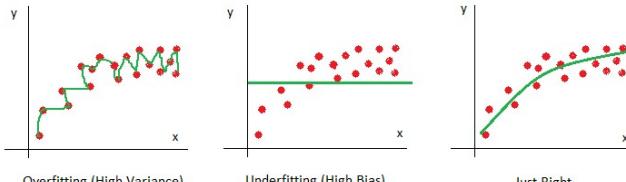
הרבבה פעמים מציאת המינימום של פונקציית המחויר היא משימה קשה. דרך מוגבלת להתמודד עם חישוב הפרמטרים האופטימלי היא שיטת (GD) gradient descent. בשיטה זו מתחילה מינוחש מסוים עבור הפרמטרים, וכל פעם מבצעים צעד לכיוון הגדיאנט השילוי. הגרדיאנט הוא הנגזרת של הפונקציה, והוא מגדיר את היכיוון שערוך הפונקציה עולה בו בצורה מקסימלית. אם לוקחים את היכיוון השילוי של הראדיאנט, בעצם הולכים לכיוון בו יש את הרידת היכי גזילה, וכך ניתן להגיע למינימום יש לבצע את הצעד ביכיוון הגרדיאנט (ϵ). בצד ימין נטען מהיקלעות לנוקודות אוכף, מוסיפים איבר נוקרא ($J(\theta)$) (מסומן באות ϵ). מבצעים את הגזרה ושינוי הפרמטרים באופן איטרטיבי עד נקודת עצירה מסויימת. באופן פורמלי, עבור ניחוש התחלתי θ_0 , בכל צעד יוצע הקידום באופן הבא (העדכן מתבצע באופן סימולטני עבור כל θ_{j+1}):

$$\hat{\theta}_{j+1} = \hat{\theta}_j - \frac{\partial}{\partial \theta_j} L(\hat{\theta}_j)$$

קיים זה יוצע שוב ושוב עד התכנסות לערך מסוים. כיוון שהבעיה קמורה מוגבطة שתיהה התכנסות למינימום, אך היא יכולה להיות איטית עקב צעדי עדכון דואלים או קטנים מדי. פרמטר ה- ϵ , learning rate (ϵ), קבוע את קצב ההתכנסות, אך רצוי לבחור פרמטר לא קטן מדי כדי לא להאט את ההתכנסות ולא גדול מדי כדי למנוע ההתכנסות.

3.1.3 Regularization and Cross Validation

אחד האתגרים המרכזיים של בעיית הרוגטיה (שאסר בעיות הלמידה) הוא לפתח מודל שייהה מוצלח לא ורק עבור אוסף הדוגמאות היידע (50 האימון), אלא שייהה מספק טוב גם עבור דוגמאות חדשות ולא מוכרות (קבוצת מבחן). כל מודל יכול לסביר מטהיה לשני כיוונים – Underfitting ו-Overfitting. Underfitting הוא מצב בו ניתנת הערכת יתר לכל נקודה בסיס האימון, מה שאורר מודל מסדר גבוה בעל שונות דזילה. במצב זה המודל מתאים רק לסת האימון, אך הוא לא מצליח להסביר גם נקודות חדשות. Overfitting – מודל שלא מתאים למטרת קוו מגמה המכיל מספק מידע על הדוגמאות הנתונות, ויש לו רעש חזק.



איור 3.2 – ניתנת משקל יותר לכל נקודה גורמת למצב בו המודל הוא מסדר גבוה ובעל שונות גבוהה (שמאל). – מודל בעל רעש חזק מייג בזרה מספק טובת את המדיע (אמצע). מצב מאוזן – מודל בעל שלגיאת מינימלית, המתאר בזרה טובת את המדיע, ובנוסף מנען משגיאת יתר עבור דוגמאות חדשות (ימין).

בדי להימנע מהטיות אלו, יש לבצע regularization – regularization אליז' המונע מהמודל להיות מוטה באופן הפוגע בתוצאות. לאחר הוספת האליוץ, פונקציית המחויר תהיה ב�ורה:

$$\text{Regularized Loss} = \text{Loss Function} + \text{Constraint}$$

יש מספר דרכים לבצע את ה- λ :

Ridge Regression / L2 Regularization

דרך אחת לבצע את ה- λ היא להוסיף נספח המתויחס ליריבוע הפרמטרים:

$$L(\theta) = \text{MSE}_{\text{train}} + \lambda w^T w = \frac{1}{n} \sum_{i=1}^n \frac{1}{2} (y_i - w^T x_i)^2 + \lambda \|w\|^2$$

כעת האופטימום של הביטוי הינו:

$$\hat{w} = (X^T X + \lambda I)^{-1} X^T Y$$

הוסףת האילוץ גורמת לכך שבנוסף לחיזוי מדויק של משתנה המטריה, המודל מנסה למצער את ריבוע הפרמטרים, ובכך לנכונות להקטין עד כמה שניתן את הערך של כל פרמטר וולימע מנצח בו נתונים משקל יתר לחלק מהפרמטרים. למעשה האילוץ מקטין את השנות של המודל ובכך עשוי למונע overfitting.

Lasso / L1 Regularization

דרך נוספת לבצע את regularization היא ליחס אילוץ המתיחס לערך המוחלט של הפרמטרים:

$$L(\theta) = \text{MSE}_{\text{train}} + \lambda |w| = \frac{1}{n} \sum_{i=1}^n \frac{1}{2} (y_i - w^T x_i)^2 + \lambda |w|$$

הוסףת האילוץ מכירה את סכום הפרמטרים להיות כמו שייתר קטן, כדי למצער כמה שניתן את פונקציית המחר. בפועל אילוץ זה מביא ל"רידוד משקלים" (sparse), כלומר כפה חלק מהמקדים להיות אפס, וכך למעשה יש מעין selection – בחירת הפרמטרים המשמעותיים יותר.

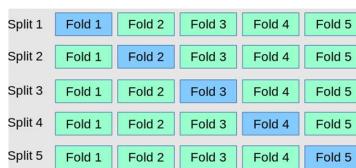
ניתן לשים לב כי עבור L1 ההשפעה של המשקלים על פונקציית המחר היא יבועית. לכן במקרה זה הרגוליזציה ת שאוף להקטין את הפרמטרים האגדולים, ובאופן כללי תנסה לדאוג לכך שכל הפרמטרים יהיו קטנים, ובאותו סדר גדול. L1 לעומת זאת שואף להקטין את כל האיברים כמו שייתר לאן קשור לאלה גדולים, ולהקטנת פרמטר מסוים מ-10-9 יש את אותה השפעה כמו הקטנה של פרמטר מ-1000-1. לכן במקרה זה הרגוליזציה תגרום לפרמטרים הפחות חשובים להתאפס, והמודל נהייה פשוט יותר.

Elastic Net

ניתן לשלב בין Ridge Regression לבין Lasso, ובכך לנכונות ל創 את המודל עבור היתרונות של כל שיטה – גם להימנע מנתנית משקל יתר לפרמטרים וגם ניסיון לאפס פרמטרים, ובכך לקבל מודל פשוט ככל הניתן. פונקציית המחר במקרה זה תהיה מהצורה:

$$L(\theta) = \text{MSE}_{\text{train}} + \lambda_1 |w|^2 + \lambda_2 |w|$$

עבור כל אחת מהדריכים, יש למציאת האופטימלי עבור הפרמטר λ (regularization) (במקרה של Elastic Net – Cross validation $\lambda = [\lambda_1, \lambda_2]$). שיטה מקובלת למציאת האופטימלי היא k-fold cross-validation – חלוקת ח-דוגמאות של סט האימון ל-k קבוצות, אימון כל תתי הקבוצות בלבד אחת, ואז בדיקת הפרמטרים שהתקבלו בשלב האימון על הקבוצה שנותרה. בכל איטרציה מוציאים חלק מסוים מהדוגמאות והופכים אותו לקבוצת מבחנים, וכך מוצאים את הפרמטר λ האופטימלי המוגן מהמשקלים להגעה מ-fitting (בדרכן כל לוקחים את הממוצע של כל ה- λ מכל האיטרציות). נפוץ להשתמש ב- $k=5$, ולמעשה עבור בדיקה טיפוסית זו יהיו 5 איטרציות, שכן אחת מהן האימון יתבצע על 80% מסט האימון, ולאחר מכן תבוצע הבדיקה של הפרמטרים שנלמדו על 20% הנדרים.



איור 3.3 Cross validation עם חלוקה ל-5 קבוצות ($k=5$). בכל פעם קבוצה אחת משמשת ל-validation (הקבוצה הכהולה).

בחירה של $n = k$ נקראת leave-one out cross validation – כיוון שלמענה בכל איטרציה יש דוגמא אחת בלבד שלא כללת בסט האימון ועליה מתבצעת הבדיקה של הפרמטרים שנלמדו.

3.1.4 Linear Regression as Classifier

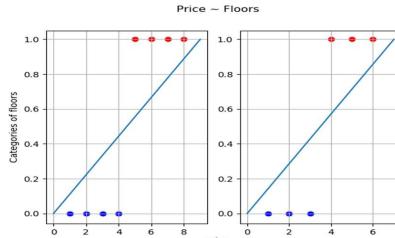
משמעות סיווג מוגדרת באופן הבא: בהינתן סט פרמטרים מסוים $\{x_1, \dots, x_n\} = X$ השיר לתכפית מסוימת, יש לסובב אותו לאחת מתוך או קטגוריות אפשריות: $\{1, \dots, m\} \in \mathcal{Y}$. לדוגמה: נתונה תמונה בעלת צו פיקסלים המייצגת חייה, יש לקבוע איזו חייה היא, כאשר הבחירה נעשית מתוך הווקטור y . לרוב הווקטור y מורכב ממספרים שלמים, שכן

אחד מהם מייצג בחרה מסוימת. בדוגמה של החיים, ניתן לקובע לדוגמא $3 = m$, כלומר $\{1, 2, 3\} \in y$, כאשר המספרים מייצגים את סט החיות $\{\text{dog}, \text{cat}, \text{chicken}\}$.

ניתן להשתמש במודל של גרסיה לינארית למשימות של סיוג. עבור המקרה של $2 = m$, יש שתי קטגוריות אפשריות, ולמעשה יש צורך לפחות נקודה אחת משתי הקטגוריות. לעומת זאת גרסיה לינארית ניתנת לביצוע מיפוי מ- \mathbb{R} ל-{0,1}, כלומר כל נקודה במרחב ממופעת לאחד משני ערכיים: קובעים ערך סוף $T = 0.5$, ועבור נקודה חדשה x_n בודקים מה היחס בין הביטוי $w^T x_n + b < 0.5$ לבין ערך הסוף. אם הנקודה החדשה מקיימת: $w^T x_n + b < 0.5$ אז הנקודה החדשה תתויג בקטgorיה 1. אחרת, הנקודה החדשה תתויג בקטgorיה 0. באופן פורמלי:

$$y = \text{sign}(w^T x_{\text{new}} + b - 0.5) = \begin{cases} 1 & w^T x_{\text{new}} + b > 0.5 \\ 0 & w^T x_{\text{new}} + b < 0.5 \end{cases}$$

בבחירה בערך הסוף $T = 0.5$ נובעת מכך שיש שתי קטגוריות {0,1}, וערך הסוף נקבע בהתאם לנקודת האמצע ביןיהם. נתונים x בתים ובער כל אחד מהם דיווח מה מחיר והאם יש בו קומה אחת או שתים.icut רצחים לבחון את היחס בין המחיר למספר הקומות ולקבוע עבור בית נתון מה מספר הקומות שלו. במקרה זה יש 2 קטגוריות: $\{1 \text{ floor}, 2 \text{ floors}\} \in y$, ויש להעיר בידע על x ובתים בכדי לבנות מודל מסווג. הדריך לעשoten את היא לבצע גרסיה לינארית, ואך שbowith מיר של בית, יש לבדוק אם $b + w^T \cdot x \geq 0.5$ או קטן ממנו, כאשר (a) הם הפרמטרים של הגרסיה הלינארית.



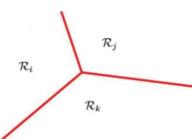
אוור 3.4 גרסיה לינארית מסווג ביאר: מיפוי הנקודות בהתחם למיקום בויש לנו ההפדרה של הגרסיה הלינארית. במקרה הימניינו ערף מתקבל עבור $x_T = 3.5 + b = 0.5$, כלומר $3.5 + b = 0.5$. במקרה השמאלי ערך הסוף מתקבל עבור $x_T = 4.5$. ערף כל בית חדש, בהינתן מחירו ניתן ישותו לאחאת משתי הקטגוריות, בהתאם ליחסו לערך הסוף.

ערף y נקודות ידועות – y_1, y_2, \dots, y_n (x_1, x_2, \dots, x_n), פונקציית המחיר הינה:

$$L(\theta) = \sum_{i=1}^n 1_{\{y_i \neq \text{sign}(w^T x_i + b + 0.5)\}}$$

הפונקציה $L(\theta)$ מכילה סט של פרמטרים – $(b, w) = \theta$. כיוון שהגזרת של הפונקציה לפי כל אחד מהפרמטרים שלה w לא תלולה רק באותו פרמטר, קשה למצאו את θ המבאים למינימום את $L(\theta)$.

ניתן להרחב את המסווג גם עבור מקרים בהם יש יותר משתי קטגוריות (multi-class). סט האימון תראה כמו במקרה הביאר, ואילו y מכילCut m קטגוריות: $\{1, \dots, m\} \in y$. במקרים אלו יש ליצור מספר קווים לינאריים, המפידים בין אזורים שונים. כדי לחשב את הקווים מבצעים התהילה'יר שנקרא all versus all, בו בכל פעם לוקחים קטגוריה אחת ובזוקים ממנה Ko ההפדרה בינה לבין שאר הקטגוריות. הפרמטרים הנלמדים של קו ההפדרה יהיו חסוט המורכב מכל הפרמטרים של הגרסיה: $\theta = \{w_1, b_1, w_m, b_m, \dots, b_m\}$.



אוור 3.5 גרסיה לינארית מרובה – הפדרה בין מספר אזורים שונים על ידי מספר קווים לינאריים.

במקרה זהה, נקודה חדשה תסואוג לקטגוריה לפי הביטוי הבא:

$$y(x) = \arg \max_i (w_1^T x + b_1, \dots, w_m^T x + b_m)$$

וכל אזכור יוגדר לפיה:

$$R_i = \{x | y(x) = i\}$$

בדומה לדוגמה הבינארית, פונקציית המבחן תהיה:

$$L(\theta) = \sum_{i=1}^n 1_{\{y_i \neq \hat{y}_i\}} s.t. \hat{y}_i = \arg \max_i (w_i^T x + b_i)$$

המושא האופטימלי יהיה וקטור הפרמטרים המביא את פונקציית המבחן למינימום:

$$\hat{\theta} = \arg \min_{\theta} L(\theta)$$

גם במקרה זהה, כיוון שהנגזרת של פונקציית המבחן לפि כל פרמטר אינה תלולה ורק באותה פרמטר, בפועל קשא למצוא את θ האופטימלי המביא את $L(\theta)$ למינימום.

3.2 Softmax Regression

3.2.1 Logistic Regression

המושא הנוצר מהאגסיה הליניארית הינו "mseog קשה" – כל דוגמא חדשה שמתבלט mseog לקטגוריה מסוימת, ואין שום מידע עד כמה הדוגמא זו דומה לקטגוריות האחרות. mseog זהה אין מספיק טוב עבור מגוון בעיות, בהן מעוניינים לדעת לא רק את הקטgorיה, אלא גם מידע נוסף על היחס בין הדוגמא החדש לבין כל הקטgorיות. לדוגמה: בהינתן מידע של גידול מסוים רוצים לדעת אם הוא מאיר או שפוי. במקרה זה ההכרעה היא לא תמיד חדיםות, ויש עניין לדעת מה הסיכוי של הגידול להיות מאיר או שפוי, שהרי יתכן שהטיפול היה שונה בין מקרה בו יש 1% שהגידול הזה הוא מסווג מסוים לבין מקרה בו יש 40% שהגידול הוא מסווג הזהה. כדי להימנע מהסיג'וג הקטgorו, יש ליצור מודל הסתברות, בו כל קטgorיה מקבלת הסתברות מסוימת. אחד המודלים הבסיסיים הינו הגרסיה לוגיסטיות (Logistic regression). עברו המושא הראשית של פונקציית הסיגמוואיד:

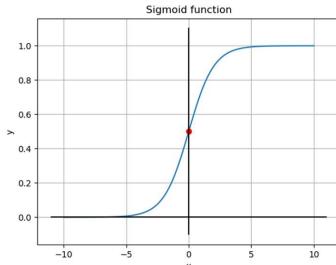
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

פונקציה זו רציפה על כל הישר, ובעצורתה ניתן להגדיר mseog עבור המקרה הבינארי:

$$p(y=1|x; \theta) = \sigma(w^T x + b) = \frac{1}{1 + e^{-(w^T x + b)}}$$

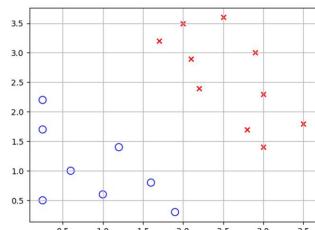
$$p(y=0|x; \theta) = 1 - \sigma(w^T x + b) = 1 - \frac{1}{1 + e^{-(w^T x + b)}} = \frac{e^{-(w^T x + b)}}{1 + e^{-(w^T x + b)}}$$

המושא לוקח את קי החלטה הליניארי, ומעברו אותו בפונקציית המחזירה ערך בטוחה [0, 1], כאשר הערך המוחזר הוא ההסתברות להיות בקטgorיה מסוימת. כדי להבין יותר טוב את משמעות המושא, יש להסתכל על גראף הסיגמוואיד:



איור 3.6 גראף הפונקציית $y = \frac{1}{1+e^{-x}}$. הנקודה (0, 0.5) מודגשת באדום.

כאשר הפונקציה b שווה בדיק-0, אזי $0.5 = \theta(x) = w^T x + b$. המשמעות של התוצאה הזו היא שאם עבור סט פרמטרים x_n מתקיים $w^T x_n + b > 0$, אז הסתברות של נקודה זו להיות משוכנת לקטגוריה 1 גוזלהחצי, לעומת ש- $w^T x + b < 0.5$. באפונסומטרי אם $w^T x + b < 0$, אז הסתברות של נקודה x_n להיות משוכנת לקטגוריה 1 קטנה מ-0.5. כתעولة השאלה מתי $w^T x + b = 0$, והתשובה היא שזה תלוי בקו הפרדה שבין הקטגוריות. לשם המכחשה נניח ונתונות מספר מדידות על שני פרמטרים - x_1, x_2 , ועבור כל נקודה (x_1, x_2) נתון גם מה הקטגוריה שהיא:



איור 3.7 דוגמא למספר מדידות התלויות בשני פרמטרים x_1, x_2 , ומישיותו לאחת משתי קטגוריות: $x \in \{blue, red\}$.

כיוון שנתנות הנקודות, ניתן ליצר בעדרתן קו רגראטי. לצורך הדוגמא נניח שיש שלושה פרמטרים והם מקיימים: $[w_1, w_2, b]^T = [1, 1, -3]$. הפרמטרים האלו מרכיבים את הקוויליארי $x_1 + x_2 = 3$, כלומר, עבור כל נקודה אם מתקיים $x_1 + x_2 > 3$ אז היא תהיה מסווגת כ-'red', אחרת היא תהיה מסווגת כ-'blue'. קו זה הוא למעשה קו הפרדה שניית בעדרתו לסוג נקודות חדשות. קו זה מקיים את המשוואה $w^T x + b = 0$, ולכן אם תהיה נקודה חדשה שางם מקיימת $w^T x_n + b = 0$, המשמעות היא שנקודה זו נמצאת בדיק על קו הפרדה. נקודה זו תקבל הסתברות של 50% להיות משוכנת לכל אחת מהקטגוריות. ככל שהנקודה החדשת מתפרק מקו הפרדה, כך הביטוי $w^T x + b$ יתרחק מ-0, ולכן גם $(w^T x + b) \sigma(w^T x + b)$ יתרחק מהערך חצי ויתקרב לאחד מערכי הקצה 0 או 1, והמשמעות היא כמובן שיש יותר סיכוי שנקודה זו שייכת לקטגוריה אחת ולא לאחרת.

כמובן שניתנו רקשת גם את המסוג ההסתברותי הזו ולהשתמש בו כמסוג קשחה: עבור דוגמא חדשה ליקחים את ההסתברויות שלה לכל אחת מהקטגוריות, ומסוגים את הדוגמא לקטגוריה בעלת ההסתברות הגבוהה ביותר. במקרה הבינארי וקטור הастברויות הינו $[x|y] = [p(y=1|x), p(y=0|x)]$, והקטגוריה של \hat{y} תהיה $\arg \max_i p(y_i|x)$ שזו \hat{y} בעל הастברות הגדולה ביותר.

3.2.2 Cross Entropy and Gradient descent

בכדי למצוא את הפרמטרים $(w, b) = \theta$ האופטימליים בהינתן n דוגמאות, ניתן להחלף את קритריון השגיאה הריבועית הממוצעת בקריטריון אחר למצער פונקציית המחר – Cross entropy. קритריון זה אומר שיש להביא למינימום את מינוס הלוג של סך הדוגמאות (הביטוי נבע משערור הנראות המרבית – Maximum likelihood –):

$$-\log P(Y|X; \theta) = -\frac{1}{n} \sum_{i=1}^n \log p(y_i|x_i; \theta) = L(\theta)$$

למעשה, יש למצוא את סט הפרמטרים $\hat{\theta}$ המביא את הביטוי למינימום: $\hat{\theta} = \arg \min_{\theta} L(\theta)$

בכדי לחשב את הביטוי יש לפתח קודם את הביטוי עבור נגזרת הסיגמאoid:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \rightarrow \frac{\partial \sigma(z)}{\partial z} = \frac{-1}{(1 + e^{-z})^2} \cdot e^{-z} \cdot (-1) = \frac{1}{1 + e^{-z}} \cdot \frac{e^{-z}}{1 + e^{-z}} = \sigma(z)(1 - \sigma(z))$$

זכור, $p(y=0|x; w, b) = 1 - p(y=1|x; \theta) = 1 - \sigma(\theta)$

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

בהתאם, הנגזרות של לוג סיגמאoid הן:

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

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

כעת יש לשים לב שהגזרת של $\log p(y=1|z)$, והגזרת של $\log p(y=0|z)$ הינה $(1 - \sigma(z))$. לכן אם $y \in \{0,1\}$, אז ניתן לרשום בקיצור: $y_i - \sigma(z) = y_i - \sigma(x_i^T w + b)$. במקרה של רוגסיה לוגיסטי, מփשים את הגזרת של $\log p(y_i|x_i; \theta)$, ולפי הפיתוח המקיים ניתן לרשום את זה כך:

$$\frac{\partial}{\partial \theta} \log p(y_i|x_i; \theta) = (y_i - \sigma(w^T x + b)) \cdot \frac{\partial}{\partial w} (w^T x + b) = (y_i - p(y_i = 1|x_i; \theta)) \cdot x_i$$

כעת לאחר הפיתוח ניתן לחזור להיבטיו $L(\theta)$ ולמציב:

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{1}{n} \sum_{i=1}^n \frac{\partial}{\partial \theta} \log p(y_i|x_i; \theta) = -\frac{1}{n} \sum_{i=1}^n (y_i - \sigma(w^T x + b)) x_i = -\frac{1}{n} \sum_{i=1}^n (y_i - p(y_i = 1|\theta; x)) x_i$$

3.2.3 Optimization

בדומה לרוגסיה לינארית, גם כאן חישוב הערך האופטימלי של $\hat{\theta}$ יהיה איטרטיבי בשיטת

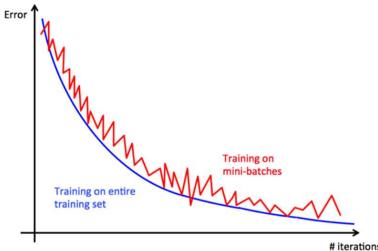
$$\hat{\theta}_{j+1} = \hat{\theta}_j - \epsilon \cdot \frac{\partial}{\partial \theta_j} L(\theta)$$

כאשר ϵ הוא הפרמטר של ה-learning rate. כיוון שפונקציית המהיר $L(\theta)$ קעורה, מובטח שתיה התכנסות ל- $\hat{\theta}$.

במקרים רבים הדadata טהו גדול, ולחשב את הגרדיינט עבור כל הדadata נדרש הרבה חישוב. בכל צעד של קידום ניתן לחשב את הוגדיינט עבור חלק מהדadata, ובצע את רקורסום לפי הכוון של הגרדיינט המתיקבל, למשל במקרה אקראי נקודה אחת ולחשב עליה את הגרדיינט. בחירה כזו נקראת Stochastic Gradient Descent (SGD), כיוון שבכל צעד שיש בחירה אקראיית של נקודה. חישוב בשיטת SGD יכול לגורם לשונות גודלה כלשהן מתקדם, ולכן לוקח מספר נקודות. חישוב הגרדיינט בשיטה זו נקרא (לעומת חישוב המתבצע על כל הדadata הנקרא batch learning). באופן פורמלי, הגרדיינט בשיטת mini-batch הוא:

$$\frac{\partial L}{\partial \theta} = \frac{\partial}{\partial \theta} \left[-\frac{1}{|V|} \sum_{i \in V} \log p(y_i|x_i; \theta) \right] \approx \frac{\partial}{\partial \theta} \left[-\frac{1}{n} \sum_{i=1}^n \log p(y_i|x_i; \theta) \right]$$

אמנם כל צעד הוא קירוב לגרדיינט, אך החישוב מאד מהיר ביחס לגרדיינט המדויק, וזה יתרון ממשוני שיש בנוסף, ניתן להוכיח שמשתנה זו מתקובל משער חסר הטה לגרדיינט האמתי.



איור 3.8 השגיאה של $\hat{\theta}$ כפונקציה של האיטרציות בשיטת gradient descent. הגרף הכהול מייצג את השגיאה בשיטת בה הגרדיינט בכל צעד מחושב על כל הדadata, והגרף האדום מייצג את השגיאה בשיטת mini-batch, בה בכל צעד הגרדיינט מחושב רק על חלק מהדadata הנבחר באופן אקראי.

בדומה ל-linear regression, גם ב-logistic regression קיימים עניין הרגולריזציה, שנועד למנוע מהמודל לתהות משקל יתר לכלי נקודה (Overfitting) או לא ליצג את הדadata בצורה מספיק טובה (Underfitting). ניתן להויסף למשל אילוץ ביחס לרכיב הפרמטר:

$$L(\theta) = -\frac{1}{n} \sum_{i=1}^n \log p(y_i|x_i; \theta) + \lambda \|\theta\|^2$$

ואז הנגזרת הינה:

$$\frac{\partial L}{\partial \theta} = -\frac{1}{n} \sum_{i=1}^n \frac{\partial}{\partial \theta} \log p(y_i|x_i; \theta) + 2\lambda \|\theta\|$$

הפרמטר λ האופטימלי מחושב על ידי ביצוע Cross validation על הדadata.

3.2.4 SoftMax Regression – Multi Class Logistic Regression

בדומה ל-linear regression, גם ב-logistic regression ניתן להרחיב את המושג גם עבור multi-class (מקורה בו יש יותר משתי קטגוריות). גם בהכללה למקרה מרובה קטגוריות יש מיפוי של כל קטגוריה להסתברות בתחום $[0, 1]$. רק כעת הפונקציה בה משתמשים היא SoftMax במקומם sigmoid. היא פונקציה המפעלת על סדרה, והיא מוגדרת כך:

$$\text{SoftMax}(z_1, \dots, z_n) = \left(\frac{e^{z_1}}{\sum_{j=1}^n e^{z_j}}, \dots, \frac{e^{z_n}}{\sum_{j=1}^n e^{z_j}} \right)$$

המונח מחשב אקספוננט בחזקת z_i , והמננה מינרמל את התוצאה, כך שscr כל האיברים לאחר הפונקציה הוא 1. במקורה בו יש מספר קטגוריות – יש מספר קווי הדרישה, וכל אחד מהם שיש סט פרמטרים θ . בהינתן נקודה חדשה, ניתן באמצעות SoftMax לחתה הסתברות לכל קטגוריה:

$$p(y = i|x; \theta) = \text{SoftMax}(w_1^T x + b, \dots, w_n^T x + b_n)$$

ואם מעוניינים לקבל סיווג קשה, ליקחים את האיבר בעל ההסתברות הגבוהה ביותר. גם במקרה זה פונקציית המחדיר תהיה cross-entropy:

$$L(\theta) = -\frac{1}{n} \sum_{i=1}^n \log p(y_i|x_i; \theta)$$

נחשב את הנגזרת של הביטוי בתוך הסכום לפי θ_i :

$$\begin{aligned} \frac{\partial}{\partial \theta_i} \log p(y_i = s|x_i; \theta) &= \frac{\partial}{\partial \theta_i} \log \frac{\exp(w_s^T x + b)}{\sum_{j=1}^n \exp(w_j^T x + b)} = \frac{\partial}{\partial \theta_i} \left(w_s^T x + b - \log \sum_{j=1}^n \exp(w_j^T x + b) \right) \\ &= 1_{\{i=s\}} x - \frac{\exp(w_i^T x + b) x}{\sum_{j=1}^n \exp(w_j^T x + b)} = (1_{\{i=s\}} - p(y = i|x)) x \end{aligned}$$

כאשר הסימון $1_{\{i=s\}}$ הינו 1 אם $i = s$ ו-0 אחרת.icut ניתן להציב את הביטוי האחרון בנגזרת של $L(\theta)$

$$\frac{\partial L}{\partial \theta_i} = -\frac{1}{n} \sum_{t=1}^n (1_{\{y_t=k\}} - p(y_t = i|x_t; \theta)) x$$

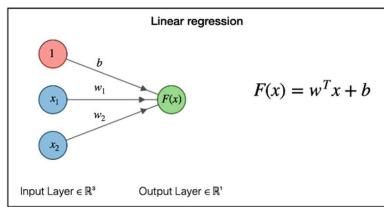
icut ניתן לחשב את θ האופטימלי בשיטת gradient descent

$$\theta_{i+1} = \theta_i - \epsilon \frac{\partial L}{\partial \theta}$$

3.2.5 SoftMax Regression as Neural Network

לשיטת logistic regression יש מספר יתרונות: היא יחסית קלה לאימון, מספקת דיוק טוב לדאטה-5טים פשועים, יציבה ל-overfitting, מיצעה סיווג הסתברותי ומתאיימה גם למקרה בו יש יותר משתי קטגוריות. עם זאת, יש לה חסרון משמעותי – קיוי ההפרדה של המודל הימם לינאריים, וזה הפדרה שאינה מפיקה טובہ עבורה בעיות מורכבות. יש מגוון בעיות בהן על מנת לבנות מודל המסוגל להפריד בין קטגוריות שונות, יש צורך במנגנון הפרדה לא לינארית.

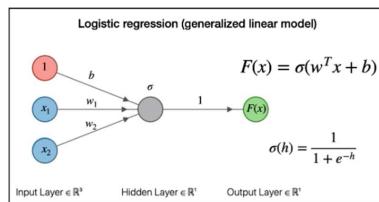
דרך מקובלת לבניית מודלים לא לינאריים היא שימוש ברששות נירונים עמודות, ובכך להבין את הקונספט שלן היבט, ראשית יש ליזג את המודלים הלינאריים כשבה של נירונים, כאשר המודל זה שקול להלוטין לכל מה שהוצע עד כה. בעיתות **Linear regression** לוחחת סט של מאפיינים ומכלילה כל אחד מהם במסקל, ולאחר מכן סוכמת את כל האלמנטים (בצירוף bias) כדי לשנהו יחד הקבוע מה הקטgorיה של סט זה. ניתן ליזג את המודל על ידי התיאור הגרפי הבא:



איור 3.9 יציג רגרסיה לינארית כרשת נירונים עם שכבה אחת.

בתיאור זה יש 2 מאפיינים המהווים את *ה-טוקן*, וכל אחד מהם מחובר למצוא בתוספת הכפלת במסקל. בנוסף יש $F(x) = w^T x + b = w_1 x_1 + w_2 x_2 + b$, וביצירוף המאפיינים המוכפלים במסקלים וה-*bias* מתקבל המוצא: $F(x)$.

כל עיגול באירור נקרא נוירון מלacional – אלמנט היכול לקלט, בצע פועל חישובית ולהוציא קלט. רגרסיה לוגיסטיבית ניתנת לתיאור באמצעות מושג דומה, כאשר הנירונים של ה-*טוקן* לא מחוברים ישירות במושא אלא עוביים דרך סיגמאיד במקורה הבנאי או דרך SoftMax במקורה בו יש יותר משתי קטגוריות:



איור 3.10 יציג רגרסיה לוגיסטיבית כרשת נירונים עם שכבה אחת.

מלבד המעבר לפונקציית הסיגמאיד, יש הבדל נוסף בין הייצוג של הרגרסיה הלינארית לייצוג של הרגרסיה הלוגיסטיבית: בעוד הרגרסיה הלינארית מספקת במצב אחד ייחד במצב אחד (מוסוג קשה), הרגרסיה הלוגיסטיבית מספקת במצב אחד וקיים באזור של מספר הקטגוריות, באופן כהו שלכל קטgorיה יש סוכבות מסוימת שה-*טוקן* שייר לאומה קטgorיה.

בפרק הבא יוצג מבנה בעל מספר שכבות של נירונים, כאשר בין שכבה לשכבה יש פונקציה לא לינארית. באופן זהה המודל שיתקבל יהיה מיפוי של סט מאפיינים באופן לא לינארי ליקטור הסתברויות במצב. הגמישות של המודל מאפשרת לתמודד עם משימות בעלות דатаה מורכב.

References

<https://www.deeplearningbook.org/>

Fitting:

<https://www.calloftechies.com/2019/08/solving-overfitting-underfitting-in-machine-learning.html>

Cross validation:

https://scikit-learn.org/stable/_images/grid_search_cross_validation.png

linear regression:

מצגות מהקורס של פרופ' יעקב גולדברג

<https://joshuagoings.com/2020/05/05/neural-network/>

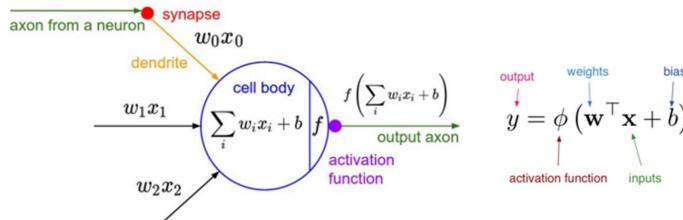
4. Deep Neural Networks

פרק זה עוסק ברשתות נירוניים עמוקות. רשת נירונים הינה חיבור של יחידות עיבוד בסיסיות (נירונים מלאכותיים) על ידי משקלים ופונקציות לא לינאריות. רשת נירונים נקראת עמוקה אם היא מכילה יותר שכבה חביה אחת. לאחר הצגת הבסיסי הרעיוני והפורמלי, יסביר כיצד ניתן לחשב את המשקלים של הרשת בצורה ישרה באמצעות המונח Computational Graph. לאחר מכן ייצאו שני תחומים העוסקים בשיפור הרשת – שיטות אופטימיזציה לתהיליך הלמידה ושיטות לבחון עד כמה המודל המתkeletal אכן מצליח בצורה טובא את הדטה עלי הוא מאומן.

4.1 Multilayer Perceptron (MLP)

4.1.1 From a Single Neuron to Deep Neural Network

ראשית יש לתאר את המבנה של יחידת העיבוד הבסיסית – נירון מלאכותי. יחידת עיבוד זו נקראת כך עקב הדמיון שלה לנירון פיזיולוגי – יחידת העיבוד הבסיסית במוח האדם האנוש. הנירון יכול לקבל מספר קלילים ולחבר אותם, אז להעביר את התוצאה בפונקציית הפעלה (activation function) שנייה בהכח לינארית. באופן סכמטי ניתן:

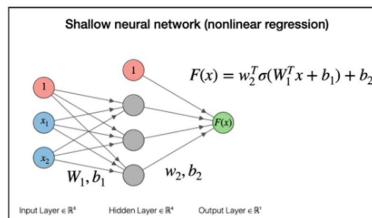


איור 4.1 ייצוג של נירון מלאכותי, המכבל קלט, סוכם אותו ומעביר את התוצאה בפונקציית הפעלה.

הקלט של הנירון הוא סט input $x \in \mathbb{R}^d$ מכפל במשקלים: $x^T w$ כאשר $w \in \mathbb{R}^d$, ואיבר קבוע b . הקלט עובר דרך סכום, ומתקבל הביטוי $y = \phi(\sum_{i=1}^d w_i x_i + b)$. לאחר מכן הסכום עבר דרך פונקציית הפעלה, ומתקבל המוצא $f(\sum_{i=1}^d w_i x_i + b)$. במקרה הפרטי בו פונקציית הפעלה היא סיגמאיד/Max-SoftMax והווצא לא מחובר לשכבה נוספת, אז למעשה מתקבלים את הרגסית הלוגיסטי.

במקרה בו הנירונים המוחברים ל-input נקראים אינטראקטיביים (interneurons), אז השכבה המוחברת ל-input נקראת שכבה חביה (hidden layer). אם יש יותר שכבה חביה אחת, הרשת מכונה רשת נירוניים عمוקה. במקרה בו יש לפחות שכבה חביה אחת, הקשר בין הכוונה למוצא אינו לינארי, וזה הינו שווי למודול. נתובן במקרה של שכבה חביה הנחשב באינטראקטיביים: סumnן את המשקלים בין הכוונה לבין השכבה החביה $w_1 \cdot b_1$, $w_2 \cdot b_2$ ואת המשקלים בין הכוונה לבין המוצא $w_1 \cdot b_1 + w_2 \cdot b_2$. ונקבל שלפחות השכבה החביה מתקבל הביטוי: $y = \phi(w_1^T x + b_1) + b_2$. ביטוי זה עבר בפונקציית הפעלה נוספת ועודף ומתקיים המוצא:

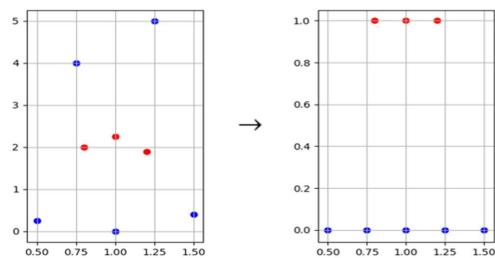
$$\hat{y} = f_2(w_2 \cdot f_1(w_1^T x + b_1) + b_2)$$



איור 4.2 רשת נירונית בעלת שכבה חביה אחת.

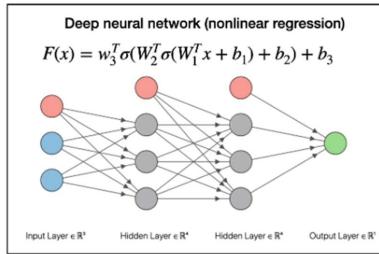
חשוב להציג שיטתית הרשת היא לבצע פעולות לא לינאריות על ה-input כך שהוא יסודר באופן חדש הנתון להפרדה לינארית. למעשה לא מבטלים את ההפרדה הלינארית הנעשית בעדרת הרגסית, אלא מבצעים לפניה שלב מקדים של העתקה לא לינארית. תהילך זה נקרא למידת ייצוגים (representation learning), כאשר בכל שכבה

מנוסים ללמידה יציג פשטוט יותר לדאטה על מנת שהוא יוכל מופרד באופן ליניארי. המיקוד של הרשת הוא אינו במשיחות סוג אלא במשיחת יציג, כך שבסתופו של דבר ניתן יהיה לՏווג את הדאטא באמצעות סיג'וֹן ליניארי פשוט (רגסיה ליניארית או לוגיסטיית).



איור 4.3 העתקה לא ליניארית של דוגמאות על ידי המשווה $\hat{y} = \begin{cases} 1, & \text{if } 3 \leq (x^2 + y^2) \leq 8 \\ 0, & \text{else} \end{cases}$. העתקה זו מאפשרת להבחין בין הדוגמאות באמצעות הפרדה ליניארית.

כארם מחברים יותר משכבה חビיה אחת, מתקבלים רשת עוקפה. החיבור בין השכבות נעשה באופן זהה – הכפלת של משקלים, סכימה והערכה בפונקציית הפעלה.



איור 4.4 רשת נירונים בעלת שתי שכבות חביות.

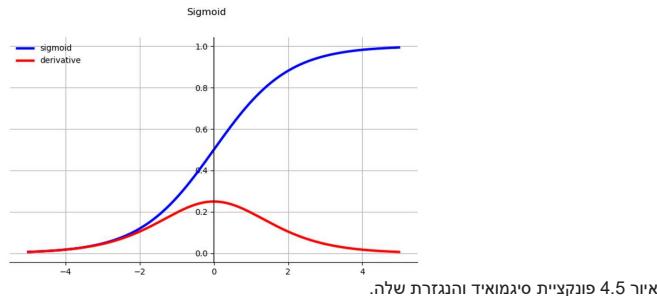
רשת נירונים בעלת לפחות שכבה חビיה אחת הינה **הינה Universal approximation**, כלומר, ניתן לציג בקירוב כל התפלגות מותנית באמצעות הארכיטקטורה זו. ככל שהרשת יותר עמוקה, כך יכולות שלה להשיג דיוק טוב יותר גדולה.

4.1.2 Activation Function
האלמנט המרכזי בכל ניירון הוא פונקציית הפעלה, ההופכת אותו לחידת עיבוד לא ליניארית. יש מספר פונקציות הפעלה מקובלות – Sigmoid, tanh, ReLU.

Sigmoid

פונקציית הסיגמויד הוצגה בפרק של רגסיה לוגיסטיית, ועתה נרחיב עליה. הפונקציה והנגזרת שלה הן מהצורה:

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \quad \frac{\partial}{\partial z} \sigma(z) = \sigma(z)(1 - \sigma(z))$$



איור 4.5 פונקציית סיגמואיד והנגזרת שלה.

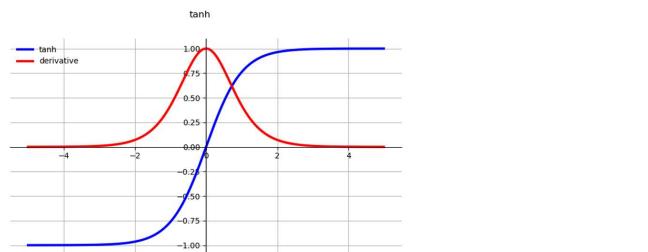
יש לפונקציה זו שלושה חסרונות:

- עבור ערכים גדולים, הנגזרת שואפת ל-0. זה כמובן יצר בעיה בחישוב הפרמטר האופטימלי בשיטת Gradient descent, שחרי בכל צעד התאפסות תלויה בградיאנט, ואם הוא מתחaes – לא ניתן לחשב את הפרמטר האופטימלי.
- הסיגמואיד לא ממורץ סיבוב -0, וזה יצא בעיה עבור דאתה שאינו מנורמל.
- הן הפונקציה והן הנגזרת דורשות חישוב של אקספוננט, ובאופן ייחודי זו פועלה יקרה לחישוב.

tanh

פונקציית טנגנס היפרבולי הינה מהצורה:

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}, \frac{\partial}{\partial z} \tanh(z) = 1 - (\tanh(z))^2$$



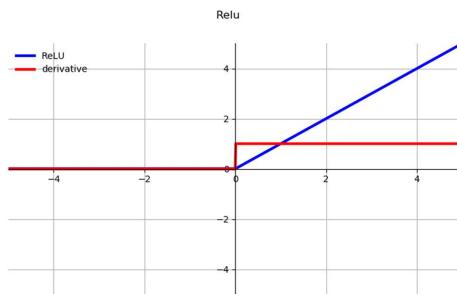
איור 4.6 פונקציית טנגנס היפרבולי והנגזרת שלה.

גם בפונקציה זו יש את הביעות של חישוב אקספוננט והתאפסות הגראדיינט עבור ערכים גדולים, אך היתרון שלו הוא שהוא ממורצת סיבוב 0.

ReLU (Rectified Linear Unit)

פונקציית ReLU מפעסת ערכים שליליים ואディשה כלפי ערכים חיוביים. הפונקציה מחזירה את המיקס'מים מבין המספר שהוא מקבלת ובין 0. באופן פורמלי צורת המשוואה הינה:

$$ReLU(z) = \max(0, z), \frac{\partial}{\partial z} ReLU(z) = 1_{\{z>0\}} = \begin{cases} 1, & z > 0 \\ 0, & z \leq 0 \end{cases}$$



איור 4.7 פונקציית ReLU והנגזרת שלה.

פונקציית ReLU עיליה יותר לחישוב מהפונקציות הקודמות, כיון שיש בה רק בדיקה של סימן המספר, אוין בה כפל או אקספוננט. בנוסף, בפונקציה זו הגרדיינט לא מתאפס בערכים גבוהים. יתרון נוסף שיש לפונקציה זו – היא מתכנסת יותר מהר מהפונקציות הקודמות (6x). לפונקציה יש שני חסרון עיקריים: היא לא ממורכחת סיבי 0, ועבור אתחול משקלים לא טוב מרכיבת הנירונים מתאפסים וזה יחסית בזבזני. כדי להתגבר על הבעיה האחורונה ניתן להשתמש בורסיות של הפונקציה, כמו למשל PReLU ו-ELU:

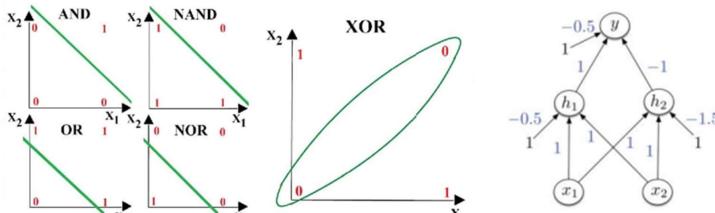
$$PReLU(x) = \max(\alpha x, x), ELU(x) = \begin{cases} x, & x > 0 \\ \alpha(e^x - 1), & x \leq 0 \end{cases}$$

בפונקציית PReLU, המקרה הפרטי בו $\alpha = 0.01$ נקרא Leaky ReLU. בפונקציית ELU, הפרמטר α הוא פרמטר נלמד.

ישן עוד פונקציות, אך אלה הן העיקריות, כאשר לרבות מקובל להשתמש בReLU ו-ELU ו/or PReLU.

4.1.3 XOR

אחד הדוגמאות הידועות ביותר שאון ניתן לפרטה לニアרית היא עליית XOR. יש שתי כניסה - x_1, x_2 והויצא הוא 0 אם הכניסות שוות ו-1 אם הן שונות. פונקציה זו מיפה שתי כניסה ליציאה, כאשר יש שתי קטגוריות במוחץ, ואין אפשרות להעביר קוויני אחד שיבחין בין הדוגמאות השונות. לעומת זאת, ניתן לבצע שלב מקדים של הפרדה לא לニアרית, ולאחריה ניתן יהה לבנות מסווג על בסיס קוו הפרדה לニアרית.



איור 4.8 אופרטור XOR ניתן להפרדה לニアרית, בשונה משאר האופרטורים הלוגיים. בעדרת ראש נירונים בעלת שכבה החניה אחת ניתן ליצור מודל שקול לאופרטור XOR.

בדוגמא המוצגת באיור הנקודות עוברות דרך שכבה החניה אחת בעלת שני נירונים - h_1, h_2 , המכילים בנוסף גם bias. פונקציית הפעולה של נירונים אלו היא פונקציית הסימן, ונitin לנתוב את המוצא של שכבה זו כך:

$$h_1 = sign(x_1 + x_2 - 0.5), h_2 = sign(x_1 + x_2 - 1.5)$$

לאחר השכבה החניה הנירונים מחוברים למוצא, שגם לו יש bias, והסכום של הכניסות והbias ממסוג:

$$y = sign(h_1 - h_2 - 0.5) = \begin{cases} 1 & \text{if } h_1 - h_2 - 0.5 > 0 \\ 0 & \text{if } h_1 - h_2 - 0.5 < 0 \end{cases}$$

נבחן את המשמעות של הנירונים: הנירון h_1 יהיה 0 אם שתי הכניסות שוות 0, אחרת הוא יהיה שווה 1. הנירון h_2 יהיה שווה 1 אם שתי הכניסות שוות 1, ובכל מקרה אחר הוא יהיה שווה 0. באופן זה לאחר השכבה החניה הראשונה,

אם גם h_1 שווה מ-0, אז יש לפחות כניסה אחת שווה, וצריך לבדוק בעזרת h_2 את המצב של הכניסה השנייה. אם גם הכניסה השנייה שווה, אז כניסה של y (יחד עם ה-bias) יתקבל מספר שלילי, ובמקרה יתקבל 0. אם הכניסה השנייה היא 0, אז $sign(0.5) = 1$. במצב בו שתי ה כניסות הן 0, יתקבל $h_1 = h_2 = 0$, וכך רק ה-bias השפיע, וכך הוא שלילי שוב ותקבל 0 ב很漂亮.

נרשום בפירוט את הערכיהם בכל שלב, עברו על הכנסיות האפשריות:

x_1	x_2	h_1	h_2	$h_1 - h_2 - 0.5$	y
0	0	0	0	-0.5	0
0	1	1	0	0.5	1
1	0	1	0	0.5	1
1	1	1	1	-1.5	0

4.2 Computational Graphs and propagation

4.2.1 Computational Graphs

כפי שהסביר לעיל, רשות ניירונות عمוקה היא רשות בעלת לפחות שכבת עמוקה אחת, ומטרתה של כל שכבה היא ללמד יוזג פשוט יותר של המידע שנכנס אליה, כך שבוטפו של דבר מסוין יהיה להבחן בין קטגוריות שונות בעקבות הפעלה לנארית. מה שקובע את השינוי של הדעתה במעברו לרשותם המשקלים ונויירונים המבצעים פעולות לא לנאריות. بعد הפעולות אותן מבצעים הנויירונים קבועות (סימנה ולאחר מכן פונקציית הפעלה), המשקלים מבקעים בהתחלה באפונן אקריא, וב误会 הדוגמאות הידועות ניתנת לאמן את הרשות ולשנות את המשקלים כך שיבצעו את למידת היוזג החדש בצוורה אופטימלית.

נכח לעשות את התהיליך הדו-שלבי הזה בעזרת Computational Graphs, שזהו למעשה גרפּי הבני מיצמתה המינימלית את התהיליך שהדאטה עבר בטור הרשות. הגרף יוכל לילץ כל רשות, וכןן באמצעות חישוב נגזרות מורכבות באופן פשוט יחסית. לאחר השלב הראשון בו מעבירים דוגמא בכל חלקו הגרף, ניתן למשל לחישוב את השגיאה הריבועית הממוצעת² ($y - \hat{y}$), להציג אותה כפונקציית המחדמי, ולמצוא את הנגזרת של כל משקל לפּי פונקציה זו – $\frac{\partial L}{\partial w_i}$; כאשר הנגזרות החלקיים מחושبات בעזרת כל הרשות.

4.2.2 Forward and Backward propagation

באותן פורמלִי, עבור A משקלים התהילה מנוסח כר:

Forward pass:

For i in 1 ... N:

Compute w_i as function of $w_0 \dots w_{i-1}$

Backward pass:

$$\overline{w_N} = 1$$

For i in $N = 1 \dots 1$:

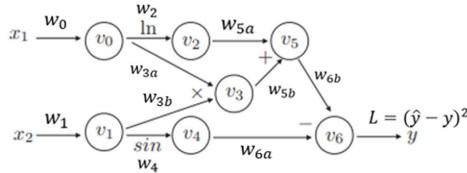
$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial w_N} \cdot \frac{\partial w}{\partial w_{N-1}} \cdots \frac{\partial w_{i+1}}{\partial w_i}$$

$$\overline{w_i} = w_i - \epsilon \frac{\partial L}{\partial w_i}$$

בשלב הראשון מחשבים כל צומת על סמך הצמתים הקודמים לו, ובשלב השני בו חוזרים אחורה, מחשבים את הנגדרת של כל משקל בעדרת כל השרשת החל מהמוצא ועד לאווט משקל, ומעדכנים את המשקל. נסתכל למשל בדוגמה הבאה:

$$y = f(x_1, x_2) = \ln(x_1) + x_1 x_2 - \sin(x_2)$$

לפונקציה זו שתי כניסה, העוברות כל אחת בנפרד דרך פונקציה לא ליניארית, ובנוסף מוכפלות אחת בשניה. באופן גרפי ניתן לראות את הפונקציה כך:



איור 4.9 הפונקציה $y = f(x_1, x_2) = \ln(x_1) + x_1 x_2 - \sin(x_2)$ מתוארת באופן גרפי.

בגרף זה יש 7 צמתים:

$$v_0 = x_1, v_1 = x_2$$

$$v_2 = \ln(v_0), v_3 = v_0 \cdot v_1, v_4 = \sin(v_1)$$

$$v_5 = v_2 + v_3$$

$$\hat{y} = v_6 = v_5 - v_4$$

לאחר שבוצע החישוב עבור \hat{y} , ניתן לחשב את הנגדרות החלקיות, בעזרת כל השרשת:

$$\frac{\partial L}{\partial w_{6a}} = -1, \frac{\partial L}{\partial w_{6b}} = -1$$

$$\frac{\partial L}{\partial w_{5a}} = \frac{\partial L}{\partial w_{6b}} \frac{\partial w_{6b}}{\partial w_{5a}} = -1 \cdot 1 = 1, \quad \frac{\partial L}{\partial w_{5b}} = \frac{\partial L}{\partial w_{6b}} \frac{\partial w_{6b}}{\partial w_{5b}} = -1 \cdot 1 = -1$$

$$\frac{\partial L}{\partial w_4} = \frac{\partial L}{\partial w_{6a}} \frac{\partial w_{6a}}{\partial w_4} = -1 \cdot (-\cos w_4) = \cos w_4$$

$$\frac{\partial L}{\partial w_{3a}} = \frac{\partial L}{\partial w_{6b}} \frac{\partial w_{6b}}{\partial w_{5b}} \frac{\partial w_{5b}}{\partial w_{3a}} = -1 \cdot 1 \cdot w_{3b}, \quad \frac{\partial L}{\partial w_{3b}} = \frac{\partial L}{\partial w_{6b}} \frac{\partial w_{6b}}{\partial w_{5b}} \frac{\partial w_{5b}}{\partial w_{3b}} = -1 \cdot 1 \cdot w_{3a}$$

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial w_{6b}} \frac{\partial w_{6b}}{\partial w_{5a}} \frac{\partial w_{5a}}{\partial w_2} = -1 \cdot 1 \cdot \frac{1}{\ln w_2}$$

המשקלים בכניסה, w_1, w_0, w , רק מעבירים ללא שינוי את הכניסות לצמתים v_1, v_0, v , ולכן הם שווים 1.

לאחר שכל הנגדרות החלקיות חושבו, ניתן לעדכן את המשקלים לפי העיקרון של GD: $w_{i+1} = w_i + \epsilon \frac{\partial L}{\partial w_i}$.

היתרון הגדול של חלוקת הרשת לגרף עם צמתים נובע מכך שאפשר כותבים את הנגדרת של $L(\theta)$ בעזרת כל השרשת, אך כל ייר בשרשראת בפני עצמו הוא חישוב פשוט ליחסוב. למשל – נגזרת של חיבור $1, 1, \dots, 1$, נגזרת של כפל היא המקדם של המשתנה לפי גוזרים, וכך באוטו אופורטוני אופורטוני שמספריים בזונת מסויים. לשיטה זו קוראים backpropagation ויהיא מאוד נפוצה ברששות עמוקות עוקב ייעילותה בחישוב המשקלים. בשונה מביעות רגיסריה, חישוב האופטימום ברששות עמוקות היא לא בעיה קמורה, ולכן תמיד יש לה בהכרח מינימום גלובלי.

עם זאת, עדכון המשקלים בשיטת Back propagation הוכיח את עצמו, למרות שהמשקלים לא בהכרח הגיעו לאופטימום שלהם.

4.2.3 Back Propagation and Stochastic Gradient Descent

כיוון שהנושא של backpropagation הוא מאוד בסיסי ברשומות נירונים, נרحب עליו את הדיבור ובסס את העקרונות המתמטיים שלו בצורה יותר عمוקה.

הקדמה – סאב גרדיאנט: נאמר ש- $V \in g$ הוא סאב-גרדיינט של פונקציה $\mathbb{R} \rightarrow V$ אם לכל $V \in \mathcal{U}$ מתקיים:

$$f(v) \geq f(u) + \langle g, v - u \rangle$$

קובץ כל הסאב-גרדיינטים של f בנקודה u מסומנת ב- $(u)f$. באופן אוטומי, $(u)f$ היא קובץ כל השרירים שנמצאים מתחת לוגרְף f בנקודה u . בפרט, עבור פונקציה f קמורה וגזירה בנקודה u יש רק אחד מתחת לוגרְף – זהה המשיק ל- f , דהיינו $(\bar{f}(u) = (\partial f)(u))$.

להוסיף איזור: Commented [ar1]:

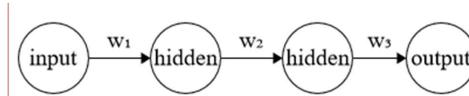
הסאב-גרדיינט משמש כהכללה למשג' הגראדיאנט כאשר אוו עוקים בפונקציות שאין בהכרח גזירות, ולעתים שימושי גם ביאטיציה של Stochastic Gradient Descent (SGD) כאשר עוקים בפונקציה שאינה גזירה אנalityית. את הסאב-גרדיינט של פונקציית מחיר l התלויה במשקולות w ומשועך בנקודה (u, x) נסמך ב- $g = l'(w, x)$.

כדוגמה קצרה לפני שמנשר, ניקח לสมן את פונקציית הערך המוחלט: $|u| = (u)_+$. לכל $0 < t$ הסאב-גרדיינט $(t)_+ = \{+1\}t = t\partial f$ וכן $t < 0$ עבור הנקודות שמקיימות $0 < t$, כלומר $\{ -1 \}t = t\partial f$. בנקודה $0 = 0$ הפונקציה אינה גזירה, אך מכל לרשותם מפושטות ש- g הוא סאב-גרדיינט של f אם מתקיים $u_0 \geq |u|$, מה שמכן רק אם $g \in [-1, 1]$ ממשום כך ותיק $[1, -1] \in (0, 0)\partial f$.

מדוע?: Commented [ar2]:

אלגוריתם backpropagation: nowog לסמן איטרציה SGD בצורתה הכללית על ידי $w^{(t+1)} = w^{(t)} - \eta \nabla_{w^{(t+1)}} g^{(t)}$, כאשר g הוא סאב-גרדיינט של פונקציית הלוּס l , דהיינו $g^{(t)}(x_t, y_t) \in \mathcal{L}$ ($w^{(t)} \in \mathcal{W}$) למן הפשטות אנו נניח כרגע ש $l = \bar{l}(g)$ (כלומר, אנו נניח שלפונקציית הלוּס יש גראדיאנט) ונתעמק בדרך החישוב המהויר של הבטיי היל', הידועה בתורו אלגוריתם backpropagation.

המסגרת הכללית שלנו היא רשת נירונים בסיסית (MLP) ופונקציית מחיר l_2 (כלומר ריבוע ההפרש בין הפלט של הרשת לבין הפלט האמתי). הצעד הראשון בתהליכי הינה להתחילה מרשת בסיסית עם שכבה אחת מירון יחיד, 2 שכבות נוספתות המכילות כל אחת נירון יחיד, וocabet פלט המכילה גם היא נירון אחד. לכל נירון (פרט לנירון הקטל) יש גם bias (b_i), וכל זוג נירונים מחוברים באמצעות משקל (alla hem w_{ij} , קר שבסר הכל, פונקציית המחריר שלו L היא פונקציה של המשנים הבאים: $L = L(w_1, b_1, w_2, b_2, w_3, b_3)$. כאמור, בוצאו של כל נירון יש פונקציית אקטיבציה (בדרכ כל לא לינארית), והמטרה הכללית היא למצוא את ערכי b , w שմבאים את השגיאה של L למינימום.



האם יש תמונה עם bias?: Commented [ar3]:

איור 4.10 רשת נירונים עם קלט יחיד, שתי שכבות חביות בעלות נירון בודד בכל אחת מהן, ומוצא בעל מירון יחיד. ככל אחד מהנירונים פרט לכינוס יש גם bias.

נתמוך בקשר בין 2 הנירונים האחרונים, כאשר נסמן את האקטיבציה של הנירון בrama $-i$ -ב- $a^{(i)}$. השגיאה של הרשת בנקודה 0 -ב- $a^{(0)}$ (למשל התמונה הראשונה מתוך 50,000 היא):

$$L_0(\dots) = (a^{(0)} - y)^2, (o \text{ is output})$$

כאשר נוכל לסמן את המוצא של האקטיבציה בrama ? באמצעות האקטיבציה של הרמה הקודמת והערכים של הנירון הנוכחי:

$$a^{(i)} = \sigma(w^{(i)}a^{(i-1)} + b^{(i)}) = \sigma(z^{(L)})$$

המטרה כעת היא להבין כמה פונקציית המחיר משתנה ביחס למשקל, כדי שוכל לעדכן את המשקלים בהתאם:

$$\frac{\partial L_0}{\partial w^{(o)}} = \frac{\partial z^{(o)}}{\partial w^{(o)}} \cdot \frac{\partial a^{(o)}}{\partial z^{(o)}} \cdot \frac{\partial L_0}{\partial a^{(o)}}$$

נחשב את הביטוי באופן מפורש:

$$L = (a^{(o)} - y)^2 \rightarrow \frac{\partial L}{\partial a^{(o)}} = 2(a^{(o)} - y)$$

$$a^{(o)} = \sigma(z^{(o)}) \rightarrow \frac{\partial a^{(o)}}{\partial z^{(o)}} = \sigma'(z^{(o)})$$

$$z^{(o)} = w^{(o)}a^{(o-1)} + b^{(o)} \rightarrow \frac{\partial z^{(o)}}{\partial w^{(o)}} = a^{(o-1)}$$

ולכן נקבל:

$$\frac{\partial L}{\partial w^{(o)}} = a^{(o-1)}\sigma'(z^{(o)})2(a^{(o)} - y)$$

עבור כל הדadata שלו, אנו לוקחים ממוצע משקל:

$$\frac{\partial L}{\partial w} = \frac{1}{n} \sum_{k=0}^{n-1} \frac{\partial L_k}{\partial w^{(o)}}$$

הביטוי הזה הוא רק אחת מהכニסות בגדיד'אנט אותו אנו רוצים לחשב (במודגש):

אחד מהכニסות לאן?: Commented [ar4]:

$$\nabla L = \left(\frac{\partial L}{\partial w^{(0)}}, \frac{\partial L}{\partial b^{(0)}}, \dots, \frac{\partial L}{\partial w^{(o)}}, \frac{\partial L}{\partial b^{(o)}} \right)^T$$

באופן דומה נוכל לרשום את הגדיד'אנט של איבר ה-bias:

$$\frac{\partial L_0}{\partial b^{(o)}} = \frac{\partial z^{(o)}}{\partial b^{(o)}} \cdot \frac{\partial a^{(o)}}{\partial z^{(o)}} \cdot \frac{\partial L_0}{\partial a^{(o)}}$$

רק האיבר הראשון במכפלה (הנגזרת של z לפ' b) משתנה, והיתר נשארים זהים:

$$z^{(o)} = w^{(o)}a^{(o-1)} + b^{(o)} \rightarrow \frac{\partial z^{(o)}}{\partial b^{(o)}} = 1$$

ולכן נקבל:

$$\frac{\partial L_0}{\partial b^{(o)}} = 1 \cdot \sigma'(z^{(o)}) \cdot 2(a^{(o)} - y)$$

והממוצע על פני כל הדadata:

$$\frac{\partial L}{\partial b} = \frac{1}{n} \sum_{k=0}^{n-1} \frac{\partial L_k}{\partial b^{(o)}}$$

כל הפיתוח שעשינו מתייחס רק לבירון של הפלט, וחישבנו כיצד פונקציית המחיר מושפעת ממשנים בקלטים של נירון זה. נרצה להכליל את הביטויים גם עבור יתר השכבות הראשונות. נתבונן למשל על השכבה אחורית לפ' אחרונה:

$$\frac{\partial L_0}{\partial w^{(o-1)}} = \frac{\partial z^{(o-1)}}{\partial w^{(o-1)}} \cdot \frac{\partial a^{(o-1)}}{\partial z^{(o-1)}} \cdot \frac{\partial L_0}{\partial a^{(o-1)}}$$

שני האיברים הראשונים ניתנים לחישוב באופן מפורש:

$$\frac{\partial z^{(o-1)}}{\partial w^{(o-1)}} = a^{(o-2)}$$

$$\frac{\partial a^{(o-1)}}{\partial z^{(o-1)}} = \sigma'(z^{(o-1)})$$

האיבר האחרון במכפלה ניתן לחישוב באמצעות כל השרשרת:

$$\frac{\partial L_0}{\partial a^{(o-1)}} = \frac{\partial z^{(o)}}{\partial a^{(o-1)}} \cdot \frac{\partial a^{(o)}}{\partial z^{(o)}} \cdot \frac{\partial L_0}{\partial a^{(o)}} = w^{(o)} \cdot \sigma'(z^{(o)}) \cdot 2(a^{(o)} - y)$$

פרט לביטוי $\frac{\partial z^{(o)}}{\partial a^{(o-1)}}$, את היתר חישבנו בשלבים קודמים, וכיון שביתוי זה שווה בדיק ל- $L^{(o)}$, ככל לרשום:

$$\frac{\partial L_0}{\partial w^{(o-1)}} = a^{(o-2)} \cdot \sigma'(z^{(o-1)}) \cdot w^{(o)} \cdot \sigma'(z^{(o)}) \cdot 2(a^{(o)} - y)$$

אם נסח זאת במילים – לצורך החישוב של $\frac{\partial L_0}{\partial w^{(o-1)}}$ הינו צריכים לדעת את $\frac{\partial L_0}{\partial a^{(o-1)}}$, ואף הוא נתן לנו על ידי החישובים שביצענו באיטרציה הקודמת, אוו' למעשה המשמעות של backpropagation.

כפי ? Commented [ar5]

עד הנה התייחסנו לרשת נוירונים בה בכל שכבה יש נוירון אחד. מלבך האינדקס הולילו שיש לכל איבר את הדיוון גם למקרים בהם יש שכבות אם יותר מנוירון אחד. בנוסף האינדקס הולילו שיש לכל איבר (המייצג את השכבה), נוסיף לכל איבר עוד משתנה (sub script) שייצג את מספר הנוירון באותה שכבה. הביטוי של L_0 יוחשב דומה, אלא שכעת יש לקחת בחשבון את כל הנוירונים בرمמה الأخيرة (נניח שיש n אלה):

$$L_0 = \sum_{j=0}^{n_o-1} (a_j^{(o)} - y_j)^2$$

כעת נסמן את המשקל בין $a_k^{(o-1)}$ ו- $a_j^{(o)}$ ב- $w_{jk}^{(L)}$. בהתאם, כל אקטיבציה תהיה מוגדרת כך:

$$a_j^{(o)} = \sigma(w_{j,0}^{(o)} a_0^{(o-1)} + \dots + w_{j,n_o-1} \cdot a_{n_o-1}^{(o-1)} + b_j^{(o)}) = \sigma(z_j^{(o)})$$

נשים לב שהמשקלים מייצגים את כל הצלעות בין הנוירון $-j$ בرمמה $-o$ לבין כל הנוירונים שברממה הקודמת – $(o-1)$:

$$z_j^{(o)} = \sum_{k=0}^{n_{o-1}-1} w_{jk}^{(o)} a_k^{(o-1)} + b_j^{(o)}$$

בשלב זה כל השרשרת יראה כך:

$$\frac{\partial L_0}{\partial w_{jk}^{(o)}} = \frac{\partial z_j^{(o)}}{\partial w_{jk}^{(o)}} \cdot \frac{\partial a_k^{(o)}}{\partial z_j^{(o)}} \cdot \frac{\partial L_0}{\partial a_k^{(o)}}$$

כאשר:

$$\frac{\partial z_j^{(o)}}{\partial w_{jk}^{(o)}} = a_k^{(o-1)}, \quad \frac{\partial a_k^{(o)}}{\partial z_j^{(o)}} = \sigma'(z_j^{(o)}), \quad \frac{\partial L_0}{\partial a_k^{(o)}} = 2 \cdot (a_k^{(o)} - y_j)$$

לכן בסך הכל קיבל שعبור דוגמה בודדת, הנגזרת של פונקציית המחיר ביחס למשקלים בرمמה o הינה:

$$\frac{\partial L_0}{\partial w_{jk}^{(o)}} = a_k^{(o-1)} \cdot \sigma'(z_j^{(o)}) \cdot 2(a_k^{(o)} - y_j)$$

וכאשר ממצאים את הנגזרת עבור a דוגמאות:

$$\frac{\partial L}{\partial w_{jk}^{(o)}} = \frac{1}{n} \sum_{i=0}^{n-1} \frac{\partial L_i}{\partial w_{jk}^{(o)}}$$

עבור ה-bias

$$\frac{\partial L_0}{\partial b_j^{(o)}} = \frac{\partial z_j^{(o)}}{\partial b_j^{(o)}} \cdot \frac{\partial a_j^{(o)}}{\partial z_j^{(o)}} \cdot \frac{\partial L_0}{\partial a_j^{(o)}} = 1 \cdot \sigma'(z_j^{(o)}) 2(a_j^{(o)} - y_j)$$

$$\frac{\partial L}{\partial b_j^{(o)}} = \frac{1}{n} \sum_{i=0}^{n-1} \frac{\partial L_i}{\partial b_j^{(o)}}$$

זכור אנו רוצים לשומר גם את $\partial L_0 / \partial a_k^{(o-1)}$ ולשם כן צריך לסכם:

$$\frac{\partial C_0}{\partial a_k^{(o-1)}} = \sum_{j=0}^{n_L-1} \frac{\partial z_j^{(o)}}{\partial a_k^{(o-1)}} \cdot \frac{\partial a_j^{(o)}}{\partial z_j^{(o)}} \cdot \frac{\partial L_0}{\partial a_j^{(o)}}$$

Commented [ar6]: זה קצת מנותק הקשר. כדי להרחב טיפה: **למה זו קשורה**

שבהרי בשונה מהמקירה בו יש רק קדקוד אחד, התלות של L_0 באקטיבציה של $a_k^{(o-1)}$ היא ביטוי של כל הנירונים המוחברים אליה בשכבה הבאה, ולא רק לאחד.

נסכם את הכל באלגוריתם:

לעדכו המשקל של השכבה ה- l :

$$\frac{\partial L_0}{\partial w_{jk}^{(l)}} = a_k^{(l-1)} \cdot \sigma'(z_j^{(l)}) \cdot \frac{\partial L_0}{\partial a_j^{(l)}}, \quad \text{average} \rightarrow \frac{\partial C}{\partial w_{jk}^{(l)}} = \frac{1}{n} \sum_{i=0}^{n-1} \frac{\partial L_i}{\partial w_{jk}^{(l)}}$$

לעדכו bias של השכבה ה- l :

$$\frac{\partial L_0}{\partial b_j^{(l)}} = \sigma'(z_j^{(l)}) \frac{\partial L_0}{\partial a_j^{(l)}}, \quad \text{average} \rightarrow \frac{\partial L}{\partial b_j^{(l)}} = \frac{1}{n} \sum_{i=0}^{n-1} \frac{\partial L_i}{\partial b_j^{(l)}}$$

זאת כאשר:

$$\frac{\partial L_0}{\partial a_j^{(l)}} = \begin{cases} \sum_{j=0}^{n_{l+1}-1} w_{jk}^{l+1} \cdot \sigma'(z_j^{l+1}) \cdot \frac{\partial L_0}{\partial a_j^{(l+1)}}, & l < L \\ 2(a_j^{(L)} - y_j), & l = L \end{cases}$$

הפלט בסופו של דבר הוא הווקטור שכנסותיו הם

$$\frac{\partial L}{\partial w_{jk}^{(l)}} = \frac{1}{n} \sum_{i=1}^{n-1} \frac{\partial L_i}{\partial w_{jk}^{(l)}}$$

$$\frac{\partial L}{\partial b_j^{(l)}} = \frac{1}{n} \sum_{i=1}^{n-1} \frac{\partial L_i}{\partial b_j^{(l)}}$$

4.3 Optimization

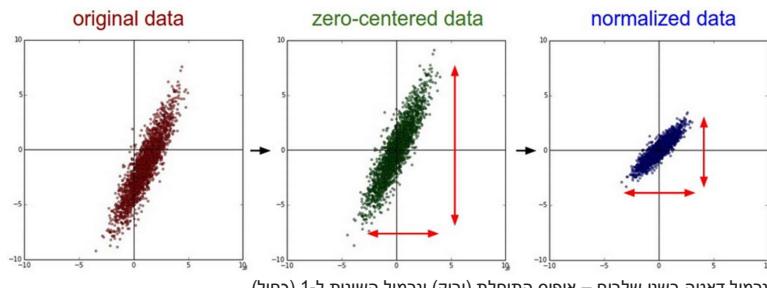
מציאת אופטימום למשקלים על פני כל העומק של הרשת היא בעיה לא קמורה, ולכן אין לה בהכרח מינימום גלובלי. לכן מלבד עדכון המשקלים בשיטת backpropagation יש לבצע אופטימיזציה נוספת על הרשת על מנת לשפר את הביצועים שלה.

4.3.1 Data Normalization

חלק מפונקציות הפעולה אינן ממורכחות סביבה-0, ועבור ערכים גבוהים הן קבועות בקריבת הגודל-אינט אל מול מתאפס, דבר שאינו מאפשר לעדכן את המשקלים בשיטת GD. כדי להמנע מהגעה לתוךם ה"זוויה" בו הגודל-אינט מתאפס, ניתן לנרמל את הדadata כך שהיא בעל תוחלת 0 ושותות 1, ובכך הוא יהיה ממורץ סביבה-0:

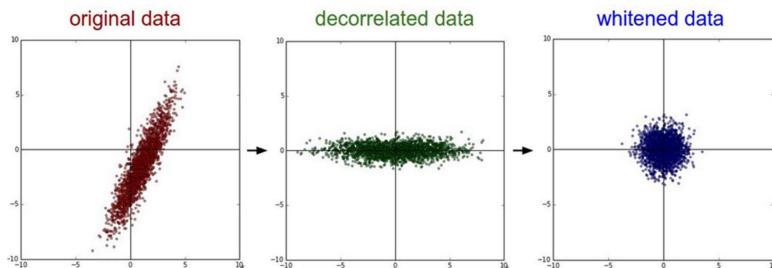
$$X_i = \frac{X_i - \mu_i}{\sigma_i}$$

ובאופן חזווי:



איור 4.11 נרמול נתונים בשני שלבים – איפוס התוחלת (ירוק) ונרמול השונות ל-1 (כחול).

שלב זה הוא למעשה שלב pre-processing הנועד להכנין את הדadata לפני כניסה לרשת, בכך לשפר את אימונו הרשת. ישנו אופנים נוספים לנרמל את הדadata – ללכון את מטריצת-h Covariance של הדadata או להפוך אותה למטריצת היחידה:



איור 4.12 דרכי נוספים לנרמל את הדadata – ללכון את מטריצת-h covariance (ירוק) או להפוך אותה למטריצת היחידה (כחול).

4.3.2 Weight Initialization

ענין נוסף שיכל להשפיע על האימון וניתן להתייחס אליו עוד בשלב ה-preprocessing ה-*initialization* המשקלים. אם כל המשקלים מאותחלים ב-0, אז המוצא וכל הגודל-אינטיהם יהיו גם כן 0, ולא יבוצע עדכון למשקלים. לכן יש לבחור את המשקלים ההתחלתיים בצורה מושכלת, למשל, להציג אותם מהתפלגות מסוימת שתאפשר אימון טוב של הרשת.

אפשרות אחת לאותחול היא להציגו עבור כל משקל ערך קטן מהתפלגות נורמלית עם שונות קטנה – $(\alpha, 0)$, כאשר $\alpha = 0.01$ or 0.1 . אותחול באופן זהה עובד טוב לרשנות קטנות 'חסין', אך ברשנות עם הרבה שכבות אותחול בערכים קטנים וורם לאיפוס הגודל-אינט מהר מאד. כדי להתמודד עם בעיה זו, ניתן לבחור $\alpha = 1$, אך זה יכול לגרום להתקדרות הגודל-אינט. שיטה שימושית יותר נקראת Xavier Initialization, הלוקחת בחשבון את הגודל של השכבות – האותחול יבוצע בזרת התפלגות נורמלית, אך השונות לא תהיה מספור ללא משמעות, אלא תהיה תלויות במספר

השכבות – $\alpha = \frac{1}{\sqrt{n}}$. שיטה זו טובת גם לרשומות עם הרבה שכבות, אך היא בעייתית במקרים בו פונקציית הפעלה הינה ReLU, כיוון שהאתחול מניח שפונקציית הפעלה ממורכצת סיבי 0 (כמו למשל \tanh). כדי לאפשר גמישות גם מבחינת פונקציית הפעלה, ניתן לבחור $\sqrt{\frac{2}{n}} = \alpha$, ואז האתחול יתאים גם ל-ReLU.

Xavier-Initialization מאפשרת לאתחול הפרמטרים היא להציג מהתפלגות אחדה, כאשר באופן דומה ל- LeCun גם כאן הgebenות יהיו תלויות בגודל השכבות – $U \left[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}} \right]$.

4.3.3 Batch Normalization

כאשר מבצעים חישובים Data normalization, למשה דואגים לכך שבכיסיה לרשות הדטה יהיה מוגרמל סיבי 0. באופן זה נמנעים מהגעה למצב בו יש ערכים גבוהים בעומק הרשת, האורמים להסתפסות או להתקדרות של הגרדיינט. בפועל, הגרמיולן מוספק טוב עבור כל השכבות, ואחריו כמו שכבת הסכלה במשקלים ומעבר בפונקציות הפעלה הרבה פעמים מותקנים ערכים גבוהים. באופן דומה $\text{Batch normalization}$ המבוצע לפני האימון, ניתן כדי האימון לבצע $\text{Batch normalization}$ שדואג לנורמל הערכים שנכנסים לניירונים בשכבות החבויות. התהיליך נעשה בשלושה שלבים:

- א. עברו כל ניירון בעל פונקציית הפעלה לא לニアרית, מחשבים את התוחלת והשונות של כל הערכים היוצאים ממנו.
 - ב. מוגרמלים את כל היציאות – מחסירים מכל יציאה את התוחלת ומחלקים אותה בשונות (בתוספת אפסיון, כדי להימנע מחולקה ב-0).
 - ג. הגרמיולן יכול לזרום לאיבוד מידע, ולכן מבצעים ליציאה המוגרמלה scale and shift – הגדזה ושינוי קנה המידה. התיקון מתבצע באמצעות פרמטרים נלמדים.
- עבור שכבות גדולות חישוב התוחלת והשונות יקר כיוון שלኒירון יש הרבה יציאות, לכן לוקחים רק חלק מהייציאות – Mini Batch: $\mathcal{B} = \{x_1 \dots m\}$

באופן פורמלי ניתן לנசח את ה-Batch Normalizing transform (Mini) כך:

$$\mu_{\mathcal{B}} = \frac{1}{m} \sum_{i=1}^m x_i, \sigma_{\mathcal{B}}^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2$$

$$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$$

$$y_i = \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i)$$

כאשר β, γ הם פרמטרים נלמדים (מעבר כל ניירון יש פרמטרים שונים).

בשלב המבחן, השונות והתוחלת שבעזרתם ממבצעים את הגרמיול אין נלקחים מהייציאות של הניירונים, אלא לוקחים ממוצע של כמה מה- Mini Batch האחרונים.

יש כמה יתרונות לשימוש $\text{Batch normalization}$: האימון נעשה מהר יותר, יש פחות ריגושים לאתחול של המשקלים, אפשר שימוש learning rate-ב- mini-batch גדול יותר (מוניון מהגרדיינט להתקדר או להאטפס), אפשר שימוש במגוון פונקציות הפעלה (גם במקרה שאין ממורכחות סיבי 0) ומוספק באופן חלקי גם רגולרייזציה (שונות נמוכה בפועל).

4.3.4 Mini Batch

במקרים רבים הדטה-ט גדול, ולהסביר את הגרדיינט עבור כל הדטה נדרש הרבה חישוב. בכל צעד של קידום ניתן לחשב את הגרדיינט עבור חלק מהדטה, וביצוע את הקידום לפי הכוון של הגרדיינט המתkeletal, למשל, ניתן לבחור באופן אקראי נקודת אחת ולחשב עליה את הגרדיינט. בחירה כזו נקראת (SGD) Stochastic Gradient Descent (SGD), כיוון שבכל צעד יש בחירה אקראיית של נקודת. בחירה קראיאית של נקודת בודדת יכולה לגרום לשונות גדולות ככל שהחישוב מתקדם, ולכן בדרך כלל מבצעים mini-batch learning – חישוב הגרדיינט על חלק מהדטה. באופן זה גם יש הפחיתה של כמות החישובים, וגם אין שונות גבוהה. אם מבצעים את החישוב בשיטה זו יש לדאוג

שהדעתה מעורבב כדי שהמשקלים אכן יתעדכו בצורה נכונה, ובנוסף שה-mini-batch היה מספק גדול קר שיהיה בו יציג כל הדעתה. כל מעבר עלי פנוי כל הדעתה-5ט נקרא Epoch (אם הדעתה הוא בגודל N, והגודל של כל-mini-batch הוא S, אז כל Epoch, או N/S איטרציות).

אמנם כל צעד הוא קירוב לגראדיינט, אך החישוב מאד מהיר ביחס לגראדיינט המדויק, וזה יתרון משמעותי שיש לשיטה זו על פני learning batch. המשקלים שמתקובלים קרובים מאד לאלו שהיו מתקובלים באמצעות batch learning.

4.3.5 Gradient Descent Optimization Algorithms

בשיטת GD, עדכון המשקלים בכל צעד הוא: $w_{i+1} = w_i - \epsilon \frac{\partial L}{\partial w}$, כאשר ϵ הוא פרמטר שנקרא lr (Learning Rate), והוא קבוע עד כמה יש לשנת המשקל בכיוון הגראדיינט. בוגדים בעיות רגסיה, אופטימיזציית ששת נוירונים היא לרוב בעיה שנייה קמורה, لكن לא מובחנת התוכנות למיניהם הגלובליים. משום כך, אם בכל צעד הולכים יותר מדי לכיוון הגראדיינט השילילי, ניתן להתקנס ליקוויד אופך או למיניהם לקלאי, שהוא אכן יתעורר בהכרח המיניהם הגלובליים. מצד שני אם מתקדים מעט מדי לכיוון הגראדיינט, המשקל יatkוויד מתעדכן, פרמטר hr-ו נועד להגבר על בעיות אלו, שכן צרך שהוא לא יהיה דיווי מדי (אחרת תהיה התברשות של המשקלים או התוכנות למיניהם לקלאי) ושללה יהיה קטן מדי (אחרת לא תהיה התקדמות או שהיא תיה מאוד איטית). כיוון שאין ערך אבסולוטי שמתאים לכל הבעיה, יש מגוון שיטות המנסות למצאו את העדכון האופטימלי בכל צעד. יש שיטות שימושות בפרמטר משתנה – lr, adaptive lr, ויש שיטות שימושות פרמטרים אחרים לביטוי של העדכון.

Momentum

ישנם מצבים בהם יש כל מיyi פיתולים בדרך ליקוויד מינימים. במצב זה, ככל צעד הגראדיינט יפנה לכיוון אחר, וההתוכנות ליקווידת מיניהם תהיה איטית. הדבר דומה לנחל שורות לים, אך הוא לא זורם יש לא הרבה פיתולים. כדי להיאץ את התוכנות במקורה זה, ניתן לנסתות לבחון את הכיוון הכללי של הגראדיינט על סמך כמה צעדים, ולהסיק התקדמותם לכיוון זה. שיטה זו נקראה מומנטום, כיון שהיא מוחפשת את המומנטום הכללי של הגראדיינט. החישוב של המומנטום מתבצע בנוסחה רקורסיבית:

$$m_{i+1} = \mu m_i - \epsilon \frac{\partial L}{\partial w}$$

ואז העדכון הינו:

$$w_{i+1} = w_i + m_{i+1}$$

הפרמטר μ הינו פרמטר דעיכה עם ערך טיפוסי בטוחה [0.9, 0.99]. ניתן להבין את משמעותו על ידי פיתוח של עוד איבר בנוסחת המומנטום:

$$m_{i+1} = \mu m_i - \epsilon \frac{\partial L(w_i)}{\partial w} = \mu^2 m_{i-1} - \mu \epsilon \frac{\partial L(w_{i-1})}{\partial w} - \epsilon \frac{\partial L(w_i)}{\partial w}$$

ניתן לראות שככל שהולכים אחורה בצעדים, כך החזקה של m גדלה. אם $\mu < 1$, אז עם הזמן הביטוי μ^n יירזוק ויקטן, ורק תהיה פחתה השפעה לעדדים שכבר היו לפני הרבה עדכנים. תחת הנחה שהגראדיינט זהה לכל הפרמטרים, ניתן לפתח נוסחה סגורה לרקורסיבי:

$$m_{i+1} = \mu m_i - \epsilon \frac{\partial L(w)}{\partial w} = \mu^2 m_{i-1} - \mu \epsilon \frac{\partial L(w)}{\partial w} - \epsilon \frac{\partial L(w)}{\partial w} = \dots = -\epsilon \frac{\partial L}{\partial w} (1 + \mu + \mu^2)$$

הביטוי שמתקובל הוא סדרה הנדסית מתכנסת, ובסך הכל מתקובל הביטוי:

$$w_{i+1} = w_i - \frac{\epsilon \frac{\partial L}{\partial w}}{1 - \mu}$$

היעילות של המומנטום תלויות בבעיה – לעיתים היא מאייצה את התוכנות ולפעמים כמעט ואינו לה השפעה, אך היא לא יכולה להזדקק.

וריאציה של שיטת המומנטום נקראת Nesterov Momentum. בשיטה זו לא מחשבים את הגראדיינט על הצעד הקודם, אלא על המומנטום הקודם:

$$m_{i+1} = \mu m_i - \epsilon \frac{\partial L}{\partial w} (w_i + \mu m_i)$$

$$w_{i+1} = w_i + m_{i+1} = (w_i + \mu m_i) - \epsilon \frac{\partial L}{\partial w} (w_i + \mu m_i)$$

שיטת זו שבדת טוב יותר עבר בעיות קמורות, ככלור היא מצליחה להתכנס יותר טוב מאשר המומנטום הרגיל, אך היא איטית יותר.

learning rate decay

באים רשותות עמוקות בדרך כלל כדי להקטין את ה- L עם הזמן. הסיבה לכך היא שכל שמתוקדים לכיוון המינימום, יש צורך בצעדים יותר קטנים כדי להצליח להתכנס אליו ולא לזרז מסביבו מצד לצד. עם זאת, קשה לקבוע כיצד לבדוק להקטין את ה- L : הקטנה מהירה שלו תימנע הגעה לאזור של המינימום, והקטנה איטית שלו לא תעזר להתכנס למינימום כאשר מגעים לאזור שלו. ישנו שלושה סוגים נפוצים של שינוי הפרמטר:

א. שינוי הפרמטר בכל כמה Epochs. מספרים טיפוסיים הם הקטנה בחצי כל 5 epochs או חולקה ב-10 כל 20 epochs. באופן כללי ניתן לומר שאורך הלמידה של ה-validation משתפר, יש להקטין את ה- L .

ב. דעיכה אקספוננציאלית של ה- L : $e^{-kt} \cdot \epsilon_0 = \epsilon$, כאשר k הינה היפר-פרמטרים, ו- t יכול להיות צעד או epoch. דעיכה לפי $\epsilon = \frac{\epsilon_0}{1+k}$.

ג. מתרגממה להקטין את כל שמתוקדים לכיוון המינימום. באופן פורמלי, אלגוריתם Adagrad מוגדר כך:

$$w_{i+1} = w_i - \epsilon_i \frac{\partial L}{\partial w}, \epsilon_i = \frac{\epsilon}{\sqrt{\alpha_i + \epsilon_0}}, \alpha_i = \sum_{j=1}^i \left(\frac{\partial L}{\partial w_j} \right)^2$$

כאשר ϵ_0 הוא מספר קטן הנועד למנוע חלוקה ב-0. כיוון שה- α הולך וגדל, הביטוי $\frac{\epsilon}{\sqrt{\alpha_i + \epsilon_0}}$ הולך וקטן, וקצב הדעיכה הוא ביחס ישיר לקצב ההתקדמות בכיוון הגרדיינט. בכך מרווחים דעיכה של ה- L , בקצב המשתנה לפי ההתקדמות. באופן ייחודי, הדעיכה של ה- L מהירה, כיוון שההסכום $\sum_{j=1}^i \left(\frac{\partial L}{\partial w_j} \right)^2 = \alpha_i$ גדל במהירות. כדי להאט את קצב הדעיכה, יש שיטות בהן נתונים יותר משקל לאירועים האחראים ופחות לאירועים שכבר עברו זמן. השיטה הפופולרית נקראת moving RMSprop, ובשיטה זו במקום לסכום את ריבוע הגרדיינט של כל הצעדים הקודמים באופן שווה, מבצעים average, וכל שעבורו יותר צעדים מסוימים עד לצעד הנוכחי, כך תהיה לו פחות השפעה על דעיכת ה- L :

$$w_{i+1} = w_i - \epsilon_i \frac{\partial L}{\partial w}, \epsilon_i = \frac{\epsilon}{\sqrt{\alpha_i + \epsilon_0}}, \alpha_i = \beta \alpha_{i-1} + (1 - \beta) \left(\frac{\partial L}{\partial w} \right)^2$$

Adam

ניתן לשלב בין הרעיון של מומנטום לבין adaptive learning rate:

$$\begin{aligned} \alpha_i &= \beta_1 \alpha_{i-1} + (1 - \beta_1) \left(\frac{\partial L}{\partial w} \right)^2, m_i = \beta_2 m_{i-1} + (1 - \beta_2) \frac{\partial L}{\partial w} \\ \hat{\alpha}_i &= \frac{\alpha_i}{1 - \beta_1^i} \hat{m}_i = \frac{m_i}{1 - \beta_2^i} \\ w_{i+1} &= w_i - \frac{\epsilon}{\sqrt{\hat{\alpha}_i + \epsilon_0}} \hat{m}_i \end{aligned}$$

מספרים טיפוסיים: $\epsilon = 10^{-2}$ or 5^{-4} . האלגוריתם למשה גם מוסיף התקדמות בכיוון המוניטום (הכוון הכללי של הגרדיאנט), וגם מביא לדעיכה אדטיבית של ה- ϵ -ז. זה האלגוריתם היל פולרי ברטשותות ענקות, אך הוא לא מושלם ויש לו שתי בעיות עיקריות: האימון הראשון הראשו לא ציבר, כיון שבתחלת האימון יש מעט נתונים לחישוב המוצע עבור i . בנוסף, המודל המתkeletal נוטה ל-*overfitting* ביחס ל-SGD עם מומנטום.

יש הרבה וריאציות חדשות על בסיס Adam שנעווד להתגבר על בעיות אלו. ניתן למשל להתחילה לאמן בקצב נמוך, וכונשר המודול מתגבר על בעיית התאיניבוט הראשונית, לאגביר את הקצב (Learning rate warm-up). במקביל, ניתן להתחיל עם Adam ולהחליף ל-SGD כאשר קритריון מסוים מתקיים. כך ניתן לנצל את ההתקנות המהירה של Adam בתחלת האימון, ואת יכולת ההכללה של SGD.

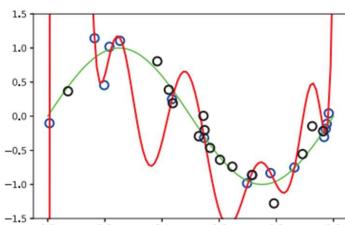
4.4 Generalization

כל מודול שנבנה נסマー על דאטה קי"ם, מטור מגמה שהמודול יתאים גם לדאטה חדש. לכן יש חשיבות גדולה שהמודול ידע להקליל כמה שיותר טוב, על מנת שהוא יתאים בaczורה טובה לא רק לדאטאות הקי"ם אלא גם לדאטא חדש. במילוי ארכיטוטור, יש לוודא שהמודול לא מונחים את הפרמטרים שלו ור' לדוגמהות שהוא זהה, אלא שינסה להגן מטור הדוגמאות מה חוקיות הכללית, שמתאימה גם לדוגמאות אחרות.

4.4.1 Regularization

ככי שהוסבר בפרק 3.1.3, מודל יול לסבול מהטיה לשני צירויות – Overfitting ו-Underfitting. Overfitting מביא בו ניתנת הערכת יתר לכל נקודה בסט האימון, מה שגורר מודל מסדר גבוה בעל שונות גודלה. במצב זה המודל מתאים רק לסט האימון, אך הוא לא מצליח להכיל גם נקודות חדשות. Underfitting הוא המצב הפוך – מודל שלא מצליח למצואו קי מגמה המכיל מספיק מידע על הדוגמאות הנוכחיות, ויש לו רושץ.

ברשותנו ניירונים, ככל שמספר הפרמטרים גדול, כך השגיאה של ה-*training* קטנה. לגבי ה-*test* השגיאה הולכת יותר עד נקודת מסוימת, ומשם היא גדלה בחזרה. בהתאם להשגואה יורדת כיוון שמלבד נתונים מודל יותר מדויק underfitting, אך בנקודת מסוימת יש יותר מדי פרמטרים והם גורמים ווותאים יותר מדי לסת האימון ומוגעים overfitting. למעשה צריך למצוא את היחס הנכון בין מספר הפרמטרים (סדר המודול) לבין גודל הדאטה.



4.13 איזור validation set בערמת training. הנקודות החקולות שיכו ל-overfitting, והשchorות שיכו ל-validation. המודול הריך לשימת זאת מתאים גם לנקודות האנשים מתאימים רק לנקודות החקולות, אך לא מאשר לומר שהוא נושא ל-overfitting.

האפשרות היכי פשיטה להימנע overfitting היא פשוטה להוריד פרמטרים, כלומר להקטין את גודל הרשת. בנוסחה
 ייתן לבצע Early stopping – לחשב בכל Epoch את גוף Loss של ה-validation, וכאשר הוא מתחילה לעלות
 להפסיך את האימון. שיטות אלה פשוטות מאוד לישום, אך ישן שיטות אחרות שמספקות ביצועים יותר טובים, ונבחן
 אותם בעת.

4.4.2 Weight Decay

לרגוליזציה של linear regression, גם ברשט מירונים ניתן להוסיף איבר ריבועי לפונקציית המחיר, מה שמכונה L2 Regularization:

$$Cost(w; x, y) = L(w; x, y) + \frac{\lambda}{2} \|w\|^2$$

ההוספה של הביטוי האחרון דוגמת לכך שהמשקל לא יהיה גדול מדי, שהרי רצויים למעורר את פונקציית המחריר, אך נשאף לכך שהביטוי הריבועי יהיה כמה שיותר קטן. בתוספת האיבר עדכון של המשקלים יהיה:

$$w_{i+1} = w_i - \epsilon \left(\frac{\partial L}{\partial w} + \lambda w \right) = (1 - \epsilon \lambda)w - \epsilon \frac{\partial L}{\partial w}$$

הביטוי הזה דומה מאוד ל-GD רגיל, כאשר נוסף איבר λw . אם $\lambda < 0$, אז ללא קשר לגרדיינט המשקל יורד בכל צעד, וזה נקרא "Weight decay".

ניתן לבצע רגולרייזציה עם איבר לא ריבועי, מה שמכונה L1 Regularization:

$$Cost(w; x, y) = L(w; x, y) + \lambda \sum_i |w_i|$$

ואז העדכון יהיה:

$$w_{i+1} = w_i - \epsilon \left(\frac{\partial L}{\partial w} + \lambda \cdot sign(w) \right)$$

בעוד L2 Regularization מושך יחיד וניסיה להקטין אותו, L1 Regularization "מעונייש" אם סכום המשקלים בערך מוחלט גדול, מה שיגרום לחלק מהמשקלים להטאפס ולידול מספר הפרמטרים של הרשת.

4.4.3 Model Ensembles and Drop Out

עבור דadata קיימים ניתן לבנות מספר מודלים, ואז כשבאים לבדוק דатаה חדש בזוקים אותו על כל המודלים ולזקחים את הממוצע. סט המודלים נקרא ensemble. ניתן לבנות מודלים שונים במספר דרכים:

א. לאמן רשת עם אתחולים שונים למשקלים.

ב. לאמת מספר רשות על חלקים שונים של הדadata.

ג. לאמן רשת במספר ארכיטקטורות.

יצירת ensemble בדרכים אלה יכולה לעזור בהכללה, אך יקר ליצור את ה-ensemble ולפעמים קשה לשלב בין מודלים שונים.

יש דרך נוספת ליצורensemble – **Dropout**, כלומר למחוק באופן אקראי נוירון אחד או יותר. אם יש רשת מסוימת ומוחקים את אחד הנוירונים – למעשה מוחקם ממנה נוירון אחד או יותר. היתרון של **Dropout** הוא שהרשותות חולקות אחת כלל פעם מוחקים ממנה נוירון אחד או יותר. היתרון של **Dropout** הוא שהרשותות חולקות את אותו פרמטרים ולבסוף מתקבלים רשות אחת מלאה עם כל הנוירונים והמשקלים. בפועל עבור כל דגימה מגרלים רשות (מוחקים כל נוירון בהסתברות $= 0.5$) וכך לומדים במקביל הרבה רשותות שונות עם אותן פרמטרים. באופן זה כל נוירון מוכחה להיות יותר משמעותי, בילאי אפשרות להסתמך על ווירונים אחרים שעשויו את הלמידה, כיוון שלא תמידם גם קיימים. אמנם כל רציה יקרה יכולה בעלת שנות גבואה אחר הממוצע של המשקלים מביא לשנות נמוכה.

בשלב המבחן, לא מפעלים את **Dropout** אלא לוחקים את כל המשקלים בחוץ. הסיבה לכך היא שבניהם להנחי שבספל האימון חצי מהഫעים המשקל היה 0 כיוון שהנוירון המקשר אליו נמחק, ובחצי מהפעעים היה משקל שנלמד. ניתן גם לחתת הסתברות אחרת למוחיקת נוירונים, למשל $= 0.25$, d , ואז שמסכים את כל הרשותות השונות יש לחלק בהסתברות המתאימה. החיסרון של שיטה זו הוא שלווח לה זמן להתקנס.

4.4.4 Data Augmentation

שיטה אחרת להימנע מ-**overfitting** היא להגדיל את סט האימון, וקר המודול שונזר יתרם ליותר דוגמאות. ניתן לעשות זאת על ידי יצירת וריאציות של הדוגמאות הקיימות. שיטה זו נקראת **Data Augmentation**, והרעיון הוא לבצע שינויים קטנים לכל דוגמא כך שהיא עדין תשמר על המשמעות המקורית שלה, אך תהיה מסוימת שונה מהמקורה בכך להיות דוגמא נוספת משמעותית בסט האימון. בדומין של תמנונות האוגומנטציות הנפוצות הן:

- סיבוב תמונה בזוית מסוימת (rotate), הבחירה מהתפלגות איחידה מהתחום $[0, 2\pi]$.
- הוספת רעש לכל פיקסל, כאשר הרעש משתנה מפיקסל לפיקסל, והוא קטן מ- ϵ .
- שינוי הגודל (rescaling) של התמונה בפקטורי מסוים – בדרך כלל הפקטור שיר לתחום $[\frac{1}{1.6}, 1.6]$.
- שיקוף התמונה (flip).
- מתיחה ומיראה של התמונה (shearing and stretching).

References

MLP:

מצגות מהקורס של פרופ' יעקב גולדברגר

<https://joshuagoings.com/2020/05/05/neural-network/>

Xor:

<https://www.semanticscholar.org/paper/Simulations-of-threshold-logic-unit-problems-using-Chowdhury-Ayman/ecd5cb65f0ef50e855098fa6e244c2b6ce02fd48>

5. Convolutional Neural Networks (CNNs)

הרשאות שתוארו עד כה הין (FC), ככלומר, כל נירון מחובר לכל הנירונים בשכבה שלפנויו וכלל הנירונים בשכבה שאחרי. גישה זו יקרה מבחינה חישובית, ופעמים רבות אין צורך בכל הקשרים בין הנירונים. לדוגמה, תמונה בגווני אפור (grayscale) בעלת 256×256 פיקסלים, המזונת לרשת FC עם $N = 1000$ נירונים, מחייבת יותר מ-65 מיליון קשרים בין נירונים, כאשר כל קשר הינו משקל המתעדכן במהלך הלמידה. קטגוריות במוח, למשל, מחייבות יותר מ-56 מיליון קשרים בין נירונים, שכן כמעט כל הקשרים והפרמטרים גולשים, אם יש מספר שכבות הרבה יותר מאשר גודל התמונה. אולם, במקרה כזה יש צורך בכל הקשרים, באופן צהוב שאלת מושג'ה מושג'ה לתחזק את הרשת. מלבד בעיית הגודל, בפועל לא תמיד יש צורך בכל הקשרים, בעוד שאלת מושג'ה מושג'ה איבר הכנסה. למשל, עבור תמונה המזונת לרשת, במשימות רבות קשור בין פיקסלים רוחקים בתמונה איננו שימושי, שכן אין חשיבות לחבר את הכנסה לכל הנירונים ורק שארם בין כל שתי שכבות סמוכות באופן מלא. כדי להימנע מעבויות אלו, לרוב יהיה כדאי להשתש בעקבות ארכיטקטורת רשתות מודרניות רבות מבוססות על שכבות קובולוציה, כל שני נירונים, אלא רק בין איברים קרובים, כפי שיפורט.

5.1 Convolutional Layers

5.1.1 From Fully-Connected Layers to Convolutions

האלמנט הבסיסי ביותר ברשתות קובולוציה היא שכבת קובולוציה, המבוצעת באמצעות פונקציית $y[n] = \sum_{m=1}^{K-1} x[n-m]w[m]$ ב כדי לקבל ייצוג אחר ופשוט יותר שלו. לרוב, שכבת קובולוציה מבוצעת פועלות קורס-קורוליזיה בין וקטור המשקלים לבין input מסוים (וקטור הכנסה או וקטור המשקלים נקרא גרעין הקובולוציה (filter) או מסנן (convolution kernel)) ובעזרתו מבוצעת פועלות הקורס-קורוליזיה הבאה:

$$y[n] = \sum_{m=1}^{K-1} x[n-m]w[m]$$

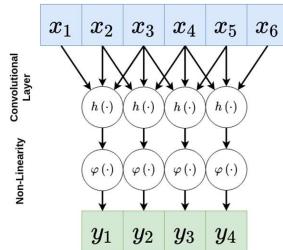
כאשר $x \in \mathbb{R}^n$ הוא וקטור הכנסה ואילו $w \in \mathbb{R}^K$ הוא וקטור המשקלים אשר נלמדים במהלך האימון. וקטור המשקלים w זהה לכל הכנסות בשכבה ולכן, מספר הפרמטרים הנלמדים לעומת רשת FC הינו קטן בהרבה – שכבת FC K משקלים ואילו שכבת קובולוציה מכילה K משקלים בלבד (לרוב מתקיים $K \ll N_{\text{inputs}} \times N_{\text{outputs}}$).

מלבד הקטנת כמות המשקלים, השימוש בגרעין קובולוציה מסייע לזרחי דפוסים ולמציאת מאפיינים. יכולות אלו נובעות מכך פועלות הקובולוציה, הבודקת חיפופה בין חלקי מוקטור הכנסה לבני גרעין הקובולוציה. הקובולוציה יכולה למצוא מאפיינים בסיגנל, וישם גרעיני קובולוציה שיכולים לבצע אוסף פועלות שימושיות, כמו למשל החלקה, נגדרת ועוד. אם מטילים על תמונה הרבה גרעינים שונים, ניתן למצוא בה כל מיני מאפיינים – למשל אם הגרעין הוא בצורה של עין או אף, אז הוא מסוגל למצוא את האזוריים בתמונה המוקריים לעין או אף.



איור 5.1 (a) קובולוציה חד ממדית בין שני פונקציות: x_1 הינו מלון בגובה 1 עם רעש טון (כחול), x_2 הינו גרעין קובולוציה מלבי שרע על פני כל הישר (כתום). פועלות הקובולוציה (שחור) בודקת את החיפופה בין הסיגナル לבני הגרעין, ובוון לראות שכן $\text{spib} = x$ $\pm 0.5 \pm 0.1$.

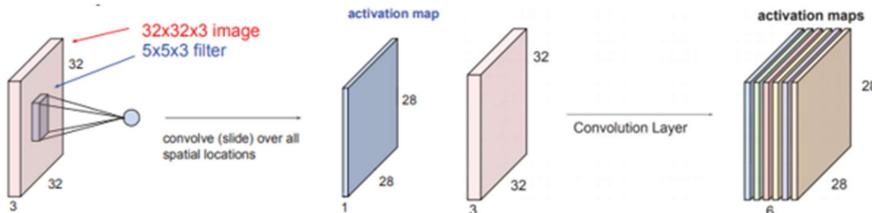
המוצא של שכבת הקובולוציה עבר בפונקציית הפעלה לא לינארית (בדרך כלל tanh או ReLU), והוא מכונה מפת הפעלה (activation map) או מפת מאפיינים (feature map). הקובולוציה יחוידת למיציאת קווי מתואר של בתר תמונה.



איור 5.2 נתונים x עובר דרך שכבה קובולוציה ולאחוריה פונקציית הפעלה, ובמוץא מתקבלת מפת אקטיבציה.

לרוב בכל שכבה קובולוציה יהיו מנסנים, אשר כל אחד מהם אמור למלוד מאפיין אחר בתמונה. ככל שהרשות הולמת ומעמיקה, כך המאפיינים בתמונה אמורים להיות מובחנים באופן חד יותר אחד מהשני, וכך המנסנים בשכבות העמוקות אמורים להבדיל בין דברים מורכבים יותר. למשל, פעמים רבות ניתן לבדוק>If המנסנים בשכבות הראשונות ייזהו את שפנות האלמנטים בתמונה או בצורות אבסוטורקטיות, ואילו מנסנים בשכבות העמוקות יותר ייזהו אלמנטים מורכבים יותר כמו איברים או חיצים שלמים בעלי צורה ידועה ומוגדרת.

הקלט של שכבה הקובולוציה יכול להיות רבע ערכיו (למשל, תמונה צבעונית המיוצגת לרוב בעזירת ערכי RGB). במקרה זה הקובולוציה יכולה לבצע פעולה על כל הערכים יחד ולסלק פלט חד ערכיו והוא יכולה גם לבצע פעולה על כל ערך ב_nfped ובירך לספק פלט רב ערכיו. גרעין הקובולוציה יכול להיות חד ממד, כפול וקשור ושפועל על קלט מסוים, אך הוא יכול להיות גם מממד גבוה יותר. לרוב, מנסנים הפועלים על תמונה הינם דו ממדיים, ופעולות הקובולוציה מבוצעת בכל שלב כפל בין המנסן לבין אחד אחר בתמונה.



איור 5.3 מסקן $F \in \mathbb{R}^{5 \times 5 \times 3}$ פועל על קלט $x \in \mathbb{R}^{32 \times 32 \times 3}$ ומטקבלת מפת אקטיבציה $y \in \mathbb{R}^{28 \times 28}$ (ימין). קלט יכול לעבר דרכם מסננים וליצור מפת אקטיבציה עם מספר שכבות – עברו שיש מהנדס המגדיל המפה הימנית $y \in \mathbb{R}^{28 \times 28 \times 6}$ (ימין).

5.1.2 Padding, Stride and Dilation

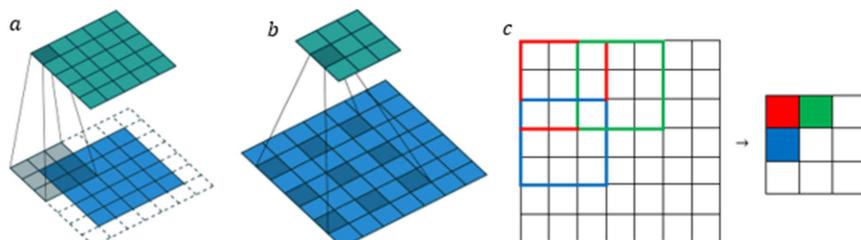
כמו ברשת FC, גם ברשת קובולוציה יש היפר-פרמטרים הנקבעים מושך וקובעים את אופן פעולות הרשת. ישנו שני פרמטרים של שכבה הקובולוציה – גודל המנסן ומספר ערוץ הקלט וכן שלושה פרמטרים עיקריים נוספיםים הקובעים את אופן פעולות הקובולוציה:

ריפור (Padding): פעולה הקובולוציה המוגדרת בעזרת המנסן הינה פעולה מוחבתה, כלומר, המנסן פועל על מספר איברים בכל פעולה. בנוסף, נדרש לבני פעולה הקובולוציה לא מוגדרת על איברי הקצאות בלבד אלא לכל הפעלת את המנסן במקומות אלו. באירוע 5.2 ניתן לראות כיצד פעולה על התמונה במדם של 32×32 מתקבלת מוקטינה את מדם הפלט – 28×28 , דבר הנובע מכך שהקובולוציה לא מוגדרת על הפיקסלים בקצוות התמונה וכן לא מופעלת עלייהם. אם רצאים לבצע את הקובולוציה גם על הקצאות, ניתן לרedef את שווי הקלט (באפסים או שפכו של ערכי הקצה), עברו מנסן בגודל $K \times K$, גודל הריפור המדרש הינו: $\text{Padding} = \frac{K-1}{2}$.

התרחבות (Dilation): על מנת לצלם עוד במספר החישובים, אפשר לפעול על אזורים יותר גדולים מתוך הנקה שערכיהם קרובים גיאוגרפית הם בעלי ערך זהה. לשם כך ניתן להרחיב את פעולה הקובולוציה תוך המשטח של עריכים קרובים. התרחבות טיפוסית הינה בעלת פרמטר d .

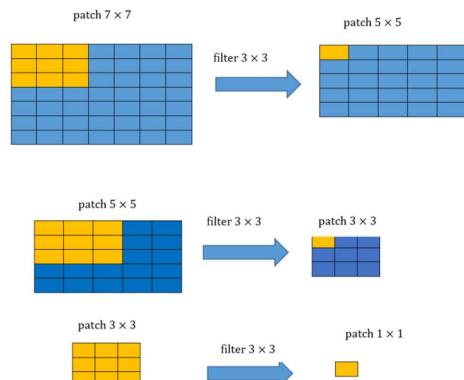
גודל צעד (Stride): ניתן להניח שלרוב הקשר המרחבי נשמר באזוריים קרובים, אך על מנת להקטין בחישובים ניתן לדלג על הפלט ולהפעיל את פעולה הקובולוציה באופן יותר דליל. לעומת זאת, אין צורך להטיל את המנסן על כל האזוריים

ההאפשרים בראשת, אלא ניתן לבצע דילוגים, כך שלאחר כל חישוב קונבולוציה יבוצע דילוג בגודל הצעד לפני הקונבולוציה הבאה. גודל עדכ טיפוס הינו 2^k .



(a) ריפוד באפסים על מנת ביצוע קובולציה גם על הקצוות של הדאטה. (b) הרחבות ($d = 2$): ביצוע הקובולציה תוך שימוש באברים סטטיסטיים מתחום הנחיה שכונראה הם דומינטיים. (c) החזת המxon בצד של $s = 2$.

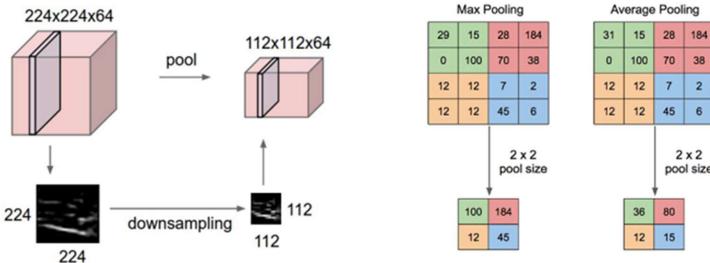
תמונה (Receptive field) של איבר בראש מוגדר להיות כל התחום בכניסה אשר משפיע על אותו איבר לאור השכבות.



אינור פיקטורי (Receptive field) של עיר מסויים במאוזן של שלוש שכבות קוגנולזיה רצואות עם מפון בגודל 3x3.

5.1.3 Pooling

פעמים רבות מרחבי מאופין בפרק ש讲 איברים קרובים דומים אחד לשני, למשל – פיקסלים סמוכים בפער אחד או יותר. ניתן לנצל עובדה זו כדי להוריד את מספר החישובים הדרוש בעזרת דילוגים (Strides) או הרחבה (dilation) כדי השאיר לעיל. שיטה אחרת לנצל עובדה זו היא לבעש Pooling – אחרי כל ביצוע קובולוציה, דגימת ערך ייחידי בעל ערכים מרוכבים, המציג את האזור. את צורת חישוב הערך של תוצאת הה-pooling ניתן לבחור בכמה דרכי, כאשר המקובלות הן בחירת האיבר הגדול ביותר באזור שלו (max pooling) או את הממוצע של האיברים (average pooling).



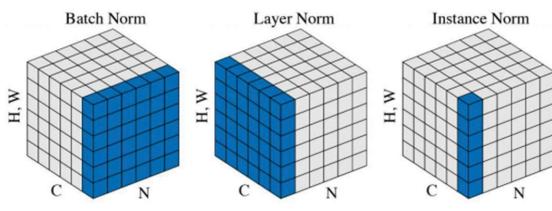
איור 5.6 הקטנת הממד של הדadata בעזרת Pooling (שמאל), והמחשה מספרית של ביצוע max/average pooling בגודל של 2×2 .

5.1.4 Training

ככל, תהליכי האימון של רשת קומבולוציה זהה לאימון של רשת FC, כאשר ההבדל היחיד הוא בארכיטקטורה של הרשת. יש לשים לב שהמשנים מופעלים על הרכבה אזורים שונים, כאשר המשנים של המשנים בכל צעד שווים, וכן אותם משקלים פועלם על אזורים שונים. לשם הפשטות נניח ויש מסנן יחיד, קלומר מטריצה אחת לממדת של משקלים. מטריצה זו מוכפלת בכל אחד מהאזורים השונים של הדadata, וכך לבצע עדכן למשקלים שלא ישקל את הגראדיאנטים של כל האזורים השונים. בפועל, הגראדיאנט בכל צעד יהיה הסכום של הגראדיאנטים על פני כל הדadata, ועבור המקרה הכללי בו יש N אזורים שונים עליהם מופעל המנסן הגראדיאנט יהיה:

$$\frac{\partial L}{\partial w_k} = \sum_{i=1}^N \frac{\partial L}{\partial w_k(i)}$$

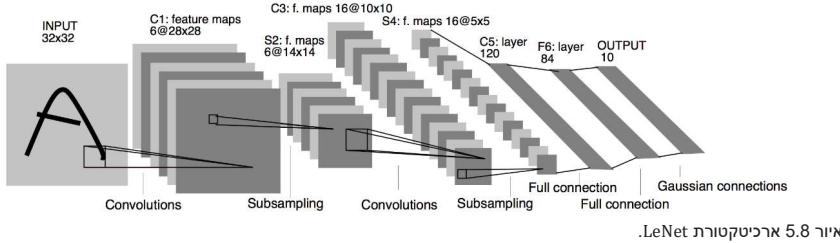
בדומה ל-FC, גם ב-CNN ניתן לבצע Batch Normalization (לשם הנוחות נתייחס לדadata כטמונה). אפשרות פשוטה ופואזה היא לנורמלizar מטען בפני עצמו על פני כמה תמונות (Batch Norm), כלומר לנקוט את כל הפיקסלים בסט של תמונות ולנורמלizar בתוחלת ובשונות שלהם. אפשרות נוספת היא לנקוט חלק מהמידע של סט תמונות, אך לנורמל אותו ביחס לאותו מידע על פני מסכנים אחרים (Layer Norm). יש וריאציות של הנורמלים האלה, כמו למשל Instance Norm, הলוקח מסנן אחד ותמונה אחת ומנורמל את הפיקסלים של אותה תמונה.



איור 5.7 נרמול שכבות של רשת קומבולוציה.

5.1.5 Convolutional Neural Networks (LeNet)

בעזרת שרשרת שכבות וחיבור כל האלמנטים השיכים לקומבולוציה ניתן לבנות רשת שלמה עבור מגוון שימושות שונות. לרוב במקומות שכבות הקומבולוציה יש שכבה אחת או מספר שכבות FC. מטרת ה-FC היא לאפשר חיבור של המידע המוליך באפיאנים שנאפסו במהלך שכבות הקומבולוציה. ניתן להסתמך על הרשת הכוללת כשי שלבים – בשלב הראשון מביצעים קומבולוציה עם מסכנים שונים, בשל דוד מהם ונדוד לדוחות מאפיין אחר, ובשלב השני מחריבים חזרה את כל המידע שנאפס על ידי חיבור כל הגינויים באמצעות FC. לראשונה השתמש בארכיטקטורה זו בשנת 1998, בראשת הנקרחת LeNet (על שם Yann LeCun, יאנן לסקון), ומוגדרת באירוויזיון 5.8. רשת זו השיגה דיקן 98.9% בזיהוי ספרות, כאשר המבנה שלו הוא שתי שכבות של קומבולוציה ושלוש שכבות FC, כאשר לאחר כל אחת משכבות הקומבולוציה מביצעים pooling.



.LeNet 5.8 ארכיטקטורת

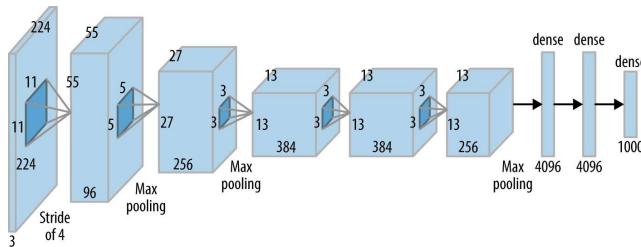
5.2 CNN Architectures

בשנים שלאחר LeNet העיסוק ברטשות נוירונים عمוקות דיבנה, עקיף חוסר המשאבים לצבע יישובים רבים בעילוי ובמהירות. בשנת 2012 רשות בשם AlexNet המבוססת על שכבות קובול羞ה ויצהה בתחרות ImageNet (תחרות ליליהו תומונות), כאשר היא הציגה שיפור משמעותי מהתוצאות היליהו טוביה בשנה שלפני. יחד עם התפתחות יכולות החישוב, העיסוק ברטשות עמוקות מרכז ופותחו הרבה מאוד ארכיטקטורות מתקרבות.

5.2.1 AlexNet

המודול הראשון באלפונטס הוא AlexNet. הנקרא גם LeNet-5. הושק ב-2012 על ידי יונתן שטרן ורמי ריבון. הושק על GPU ומשתמש במודול pooling. המודול השני הוא FC. הנקרא גם LeNet-6. משלב מודול pooling ומודול FC. המודול השלישי הוא max-pooling. הנקרא גם LeNet-7. משלב מודול max-pooling ומודול FC.

פונקציית האקטיבציה של הרטה הינה ReLU (בשונה מ-LeNet) שהשתמשה ב- \tanh ($tanh$) , והיפר פרמטרים הם: בערך 60 מיליון. Dropout=0.5 ,batch size=128 ,SGD+momentum=0.9 ,lr=1e-2 . בסך הכל מספר הפרמטרים בראש הינן



איור 5.9 ארכיטקטורת AlexNet

שנה לאחר הפרסום של רשת דומה בשם AlexNet, פותחה רשת דומה בשם ZFNet, הבנוייה באמצעות הארכיטקטורה עם הבדלים קטנים בהיפר-פרמטרים ובמספר הפליטרים: השכבה הראשונה של הקובולוציה הפכה מ- 11×11 , $s=4$ ל- 7×7 , $s=2$, ושבבות 3-4-5 מספר הפליטרים הוא 512,1024,512 בהתאם. הרשת השגה שיפור של כ- 5% על פי AlexNet. המمد של השכבות בשתי הארכיטקטורות אינו נובע מסיבה מסוימת אלא מניסוי וטעה – ניסוי תוצאות רבות ומתקן נבחרה זו בעלת הביצועים הטובים ביותר. לאחר שהרשומות מובוסות קובולוציה הוכחו את כוחן, השלב הבא היה לבנות רשותות עמוקות, ובועלות ארכיטקטורה הנשענת לא רק על ניסויים אלא גם על היוגון מסויים.

5.2.2 VGG

שנה לאחר הוצאה של ZFNet רשת עמוקה – בעלת 19 שכבות, המנצלת יותר טוב את שכבות הקונוליזציה. מפתחי הרשת הראו כי ניתן להחליף שכבת פילטרים של 7×7 בשכבות של 3×3 ולקבל את תמן (receptive field), כאשר מרויינים חסכו משמעותי במספר הfilters הנלדיים. לפילטר בגודל $d \times d$ הפעול על c ערוצי קלט ופלט יש d^2c^2 פטמררים ולמדיים, וכן לפילטר של 7×7 יש $49c^2$ פטמררים ולמדיים ואילו לשכבות של 3×3 יש $9c^2 = 3 \cdot 3^2 = 27c^2$ פטמררים נלדיים – חיסכון של 45%. הרשת המקורית שפיתחו נקראה VGG19 והיא מכילה 138 מיליון פטמררים, ויש לה וריאציה המוגיפה עוד שתי שכבות קונוליזציה ומcona VGG16.

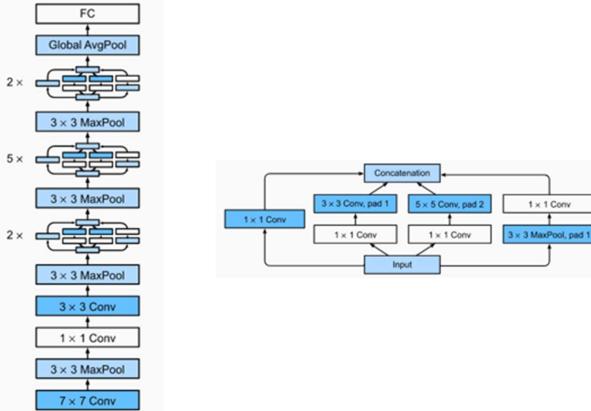


איור 5.10 ארכיטקטורת VGG (ימין) ביחס לארכיטקטורת AlexNet (שמאל).

5.2.3 GoogleNet

המודלים הקודמים היו יקריםחסיבית עקב מספר הפטמררים הגדל. בכך לייצלו להציגenos עם אותם עמוק אבל עם הרבה פחות פטמררים, הקבוצת מפתחים מוגול הציגו קונגרס module inception. בлок המבצע הרבה פעולות פשוטות במקביל, במקומות לבעלה אחת וורכמת. כל בлок מקבל input ו被执行 עליון ארבעה חישובים במקביל, כאשר הממדים של מוצאי כל הענפים שווים כך שנitin לשרשן וותם יחד. ארבעת הענפים הם: קונוליזציה 1×1 , קונוליזציה 1×1 עם padding 3×3 עם padding 1 , קונוליזציה 1×1 ולחדרה קונוליזציה 5×5 עם max pooling 3×3 , padding 1 ולחדרה קונוליזציה 1×1 . לבסוף, הפלטים של ארבעת הענפים משוחזרים יחד ומהווים את פלט הבולח.

המבנה הזה שקול למספר רשותות במקביל, כאשר היתרונו של המבנה זה הוא כפוף: כמה פטמררים נמוכה ביחס לרשותות קדומות וחישובים מהירים כיון שהם נעשים במקביל. ניתן לחבר שכבות קונוליזציה רגילים עם בЛОקים אלה, ולקבל רשת עמוקה. נעשו הרבה ניסויים כדי למצוא את היחס הנכון בין הרכבים והמדדים בכל שכבה המבאים לביצועים אופטימליים.



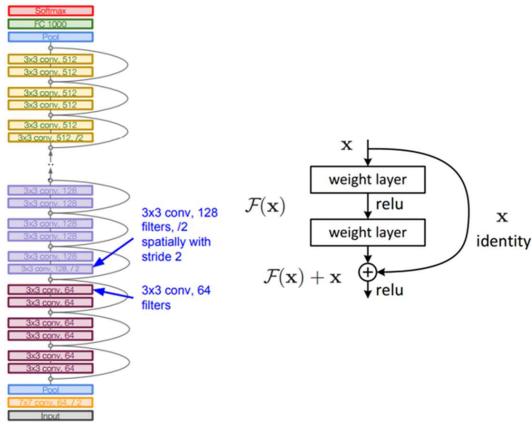
איור 5.11 Inception Block 5.11 (ימין), וארQUITקטורת GoogleNet מלאה (שמאל).

5.2.4 Residual Networks (ResNet)

לאחר שראנו שככל שהרשת عمוקה יותר כך היא מSIGGA תוצאות טובות יותר, ניסו לבנות רשתות עם מAOות שכבות, אך הן השיגו תוצאות פחות טובות מהרשתות הקודמות שהן בעלות סדר גודל של 20 שכבות. הבעיה המרכזית של הרשתות העמוקות נבעה מכך שלאחר מספר שכבות מסוים התקבל יוווג מספיק טוב, ולכן השכבות היז צריכות לא לשנות את הקלט אלא להעביר את היצוג כמו שהוא. בשביל לבצע זאת המשקלים בשכבות אלו צריכים להיות 1. הסתבר של שכבות קשה ללמידה את פונקציית החזקהות וכן למשה פגשו בתוצאה. אתגר נוסף ברשתות עמוקות נבע מהקיים לבצע אופטימיזציה כמו שצורך למשקלים בשכבות עמוקות.

ניתן לוסח את הבעיה המרכזית בכך מעת שנות – בהינתן רשת עם N שכבות, יש טעם להוסיף שכבה נוספת רק אם היא תוסיף מידע שלא קיים עד עכשוי. כדי להבטיח שישכהה תוסיף מידע, או לפחות הפחות לא פגאג במידע הקיימים, בנו רשת חדשה בעזרת Residual Blocks – יצירה בЛОקים של שכבות קובולוציה, כאשר בנוסף למעבר של המידע בתוך הבלוק, מחברים גם בין הcalcisa למוצא שלו. כעת אם בЛОק מבצע פונקציה מסוימת (x, \mathcal{F}) , אז המוצא הימן $x + \mathcal{F}(x)$. באופן זהה כל בLOCK ממקיך בלLOAD משווה שווה למורה שלמד עד עכשי, ואם אין מה להוסיף – הפונקציה \mathcal{F} פשוט נשארת 0. בנוסף, המבנה של הBLOCKים מונע מהגדיאנט בשכבות העמוקות להתבדר או להאטפס, והאימון מצליח להתכנס.

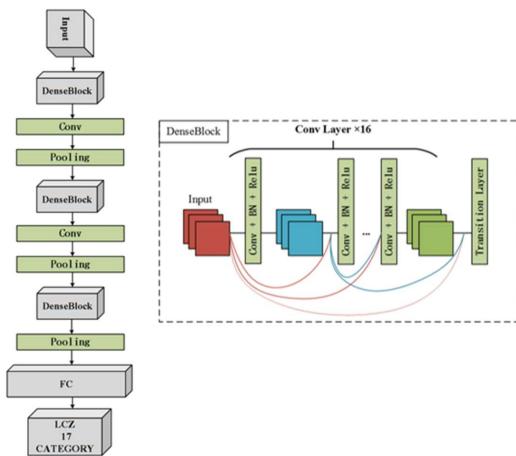
באופן זהה פותחה רשת בעלת 152 שכבות אשר הציגה ביצועים מעולים ביחס לכל שאר הרשתות באותה תקופה. השכבות היו מורכבות משלשות של BLOCKים, כאשר בכל BLOCK יש שתי שכבות קובולוציה. בין כל שלשה יש הcapella של מספר המנסנים והורדה של הממד פי שניים בעזרת pooling. ההיפר-פרמטרים הם: Batch normalization: lr=0.1, SGD+momentum=0.9, Xavier initialization weight decay=1e-5, batch size=256. רשתות מתקדמות יותר שיילבו את גישת ה-inception ייחד עם ResNet על מנת לשלב בין היתרונות של שתי השיטות.



איור 12 יחיד (ימין), וארQUITטורת ResNet Block 5.12 מלאה (שמאל).

5.2.5 Densely Connected Networks (DenseNet)

ניתן להרחיב את הרעיון של Residual Block כך שלא רק מחברים את הכניסה של כל בלוק לモץ שלו, אלא גם שומרים את הכניסה בפwi עצמה, ובודקים את היותו שלה לשכבות יותר עמוקות. Dense block הוא בלוק בעל כמה שכבות, הבניי כך שכיסה של כל שכבה מוחברת לכל השכבות אחריה. ניתן לממשו באמצעות בלוקים כאלה ייחד ולבצע בוייהם כל מיינ פועלות כמו pooling או פפיאו שכבת קונוליזציה בשמאית. ניתן שמשלבים כמה כניסה של בלוקים שונים, יש בעיה של התאמת ממדים, משומש כל בלוק מגדיל את מספר העדיצים, חיבור של כמה בלוקים יכולם ליצור מודול מורכב מז'. כדי להתגבר על בעיה זו הוספו שכבות transition בסוף כל המבצעות קונוליזיות 1×1 עם רוחב צעד = 2, ובכך מוסר העדיצים נותר סביר והמודול לא נשעה מושך מז'.

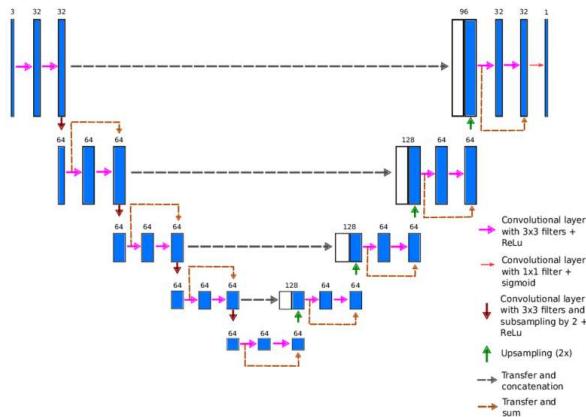


איור 13 יחיד (ימין), וארQUITטורת DenseNet מלאה (שמאל).

5.2.6 U-Net

ברשותות קונוליזיה המיועדת לטוווג, בסוף התהילה מתקבל וקטור של הסטטים, כאשר כל איבר הוא הסטטוט של label מסוים. בשימוש סגמנטציה ה בעית, כיוון שאריך בסוף התהילה לא רק ללמידה את האפיאים שבתמונה ועל פיהם לקבוע מה יש בתמונה, אלא צריך גם לשחרר מיקומו הפיסטיים והINGTONים שלהם ביחס לתמונה המקורית עם הסגמנטציה המתאימה. כדי להתמודד עם בעיה זו ציעו את ארכיטקטורת U-Net, המכילה שלושה חלקים עיקריים: כיווץ, צואר בקבוק והרחבה (contraction, bottleneck, and expansion section).

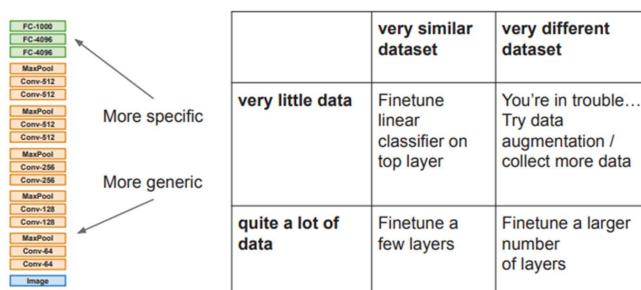
באיור, בחלק הראשון יש טופולוגיה רגילה של רשת קובולוציה, המבצעת בעזרת שכבות קובולוציה וביצוע pooling. השוני בין השלב הזה לבין רשת קובולוציה קלאסית הוא החיבור שיש בין כל שלב בתהילן לבין חלקיים בהמשך התהילן. לאחר המעבר בזאואר הבקבוק של למשה שוחרר של התמונה עם הסגנטציה. השוחרר נשען עזרת sampling על הוקטור שהתקבל בזאואר צוואר הבקבוק יחד עם המידע שנשאר מהחיק הריאון של התהילן. פונקציית המחר שמשתמשים ברשת זו נקראת cross-entropy loss, הבודקת כל פיקסל ביחס ל-amti-alio הוא שייר.



.איור 5.14 ארכיטקטורת U-Net.

5.2.7 Transfer Learning

כאשר נתקיים במשימה חדשה, אפשר להתקין עבורה ארכיטקטורה מסוימת ולאמן רשת עמוקה. בפועל זה יקר ומסובך להתאים רשת מיחודה לכל בעיה ולאמן אותה מהותהלה, ולכן ניתן להשתמש ברשתות הקיימות שאומנו כבר ולהתאים אותן לביעיות אחרות. גישה זו נקראת Transfer Learning, והางיון מהဓירה טוען שעבור כמעט כל סוג דאטה השכבות הראשונות לומדות אותו דבר (יזהו שפות, קווים וצורות כלויות, מאפיינים כלים וכו') ולכן ניתן להעתה המשמש בהן פעמים רבות ללא שינוי כל. משום כך, בפועל בדרך כלל לוקחים רשת קיימת ומחילפים בה את השכבות האחרונות או מוסיפים לה עוד שכבות בסופה, ואז מאמנים את השכבות החדשות על הדאטה החדש קר שהן תהינה מוכנות לדאטה הגרפי של המשימה החדשה. ככל שיש יותר דאטה חדש ניתן להוסיף יתר שכבות ולקבל דיוק יותר טוב, וככל שהמשימה החדשת דומה יותר למשימה המקוריית של הרשת קר יש צורך בפחות שכבות חדשות. כמו כן, משום שבשיטה זו נדרשים לאמן מספר שכבות קטן יותר, קטן ה- overfitting הנבע מחוסר דאטה.



.Transfer Learning 5.15

5. References

Convolutional:

<https://github.com/technion046195/technion046195>

מצגות מהקורס של פרופ' יעקב גולדברגר

AlexNet:

<https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaeccccc96>

VGG

<https://arxiv.org/abs/1409.1556>

GoogleNet

http://d2l.ai/chapter_convolutional-modern/vgg.html

ResNet

<https://arxiv.org/abs/1512.03385>

DenseNet

<https://arxiv.org/abs/1608.06993>

<https://towardsdatascience.com/review-densenet-image-classification-b6631a8ef803>

U-Net

<https://arxiv.org/abs/1505.04597>

6. Recurrent Neural Networks

היתרון של שכבות קונבולוציה על פני FC הוא ניצול הקשר המרחבי שיש בין איברים שונים בדата, כמו למשל פיקסלים בתמונה. יש סוג אחד בהם האיברים השונים יוצרים סדרה שיש לסדר האיברים חסיבות, כמו למשל טקסט, גל קול, רצף DNA ועוד. כמון שדעתה מהסוג זהה דרוש מודל הנוטן חסיבות לסדר של האיברים, מה שלא קיים ברשומות קונבולוציה. בנוסף, הרבה פעמים הממד של הקלט לא ייעז או משתנה, כמו למשל מספר המילים במשפט, וגם אף יש מתחת את הדעת. כדי להתמודד עם ארגונים אלו יש לבנות ארכיטקטורה שמקבלת סדרה של קיטורים ומציאת קיטור יחיד, כאשר הוקטור היחיד מוקוד קשורים על הדטה המקורי שנכנס אליו. את קיטור המוצא ניתן להעביר בשכבה FC או בכל מסוג אחר, תליי באופי המשימה.

6.1 Sequence Models

6.1.1 Vanilla Recurrent Neural Networks

רשתות רקורסיביות הן הכללה של רשתות נוירונים ע深深ות, כאשר הן מכילות משקלות המאפשרות להן לתרגם שימושות לסדר של איברי הכניסה. ניתן להסתכל על משקלות אלה כרכיב זיכרון פנימי, אשר כל איבר שנכנס משוקל ביחס לפונקציה קבועה בתוספת רכיב משתנה שתלו依 בערך העבר. כאשר נכנס וקיטור א', הוא מוכפל במשקל w_{xh} ומכנס לרכיב זיכרון h_t , כאשר h_t הוא פונקציה של x_t, h_{t-1}

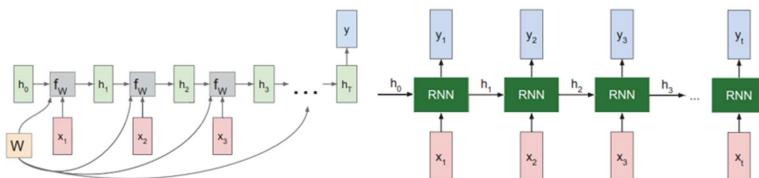
$$h_t = f(h_{t-1}, x_t)$$

מלבד המשקלים הפעילים על וקיטור הכניסה, יש גם משקלים שפועלים על המשקלות הפנימיות (רכיב הזיכרון) – w_{hh} , ומshallים הפעילים על המוצא של רכיב זה – w_{hy} . המשקלים w_{hx}, w_{hy} הם לל' השלבים, והם מתעדכנים ביחס. כמו כן, הפונקציה f_w היא קבועה לכל האיברים, למשל tanh , ReLU או sigmoid . באופן פורמלי התהילין נראה כך:

$$h_t = f_w(w_{hh}h_{t-1} + w_{xh}x_t), f_w = \tanh/\text{ReLU}/\text{sigmoid}$$

$$y_t = w_{hy}h_t$$

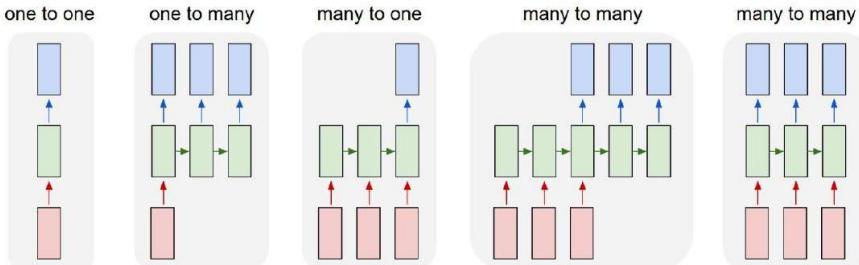
באופן סכמטי התהיליך נראה כך:



איור 6.1 ארכיטקטורות RNN בסיסיות: One to many (ימין) ו-many to one (משמאל). על כל חץ יש משקל מתאים עליי מתבצעה הלמידה.

כזכור שניית גם לשרשר שכבות ח比亚ות ולקבל רשת עמוקה, כאשר פלט של שכבה מסוימת הופך להיות הקלט של השכבה הבאה. ישנו מודלים שונים של RNN, המתאימים לביעות שונות:

- One to many – יש קלט יחיד ורוצים להוציא סדרת פלטים, למשל מכינים תמונה לרשת ורוצים משפט שיתאר אותה (Image captioning).
- Many to one – רוצים לקבוע משהו יחיד עברו קלט סדרתי, למשל מקבלים משפט ורוצים לsegue את הסנטימנט שלו – האם הוא חיובי או שלילי.
- Many to many – עברו כל סדרת קלט יש סדרת פלט, למשל תרגום משפה אחת לשפה אחרת – מקבלים משפט מוצאותים משפט.



איור 6.2 מודלים שונים של RNN.

6.1.2 Learning Parameters

הלמידה של הרשת נעשית בדומה לרשותם שבפרקם הקודמים. עבור דатаה $(x_1, \dots, x_n), (y_1, \dots, y_n)$ נגדיר את פונקציית המחרה:

$$L(\theta) = \frac{1}{n} \sum_i L(\hat{y}_i, y_i, \theta)$$

כאשר הפונקציה $L(\hat{y}_i, y_i, \theta)$ תואמת למשימה – עבור מושימות סיווג נשותמusp-by-cross entropy בקריטריון MSE. האימון יבוצע בעזרת GD, אך לא ניתן להשתמש ב-backpropagation הרגלי כיוון שכל משקל מופיע פעמיים – למשל w_{hh} פועל על כל הכניסות ו- w_{hh} פועל על כל רכיבי ה- \hat{y}_i . כדי לבצע את עדכון המשקלים משתמשים ב-(BPTT) backpropagation through time (BPTT) – מסתכלים על הרשת הנפרשת כרשת אחת גדולה, מחשבים את הגרדיאנט עבור כל משקל, ואז סוכנים או ממצאים את כל הגרדיאנטים. אם הדאטה בכינוס הוא בגודל n , קלומר יש n דגימות בזמןן, אז יש n רכיבי \hat{y}_i – n משקלים w_{hh} . לכן הגרדיאנט המשקל w_{hh} יהיה:

$$\frac{\partial L}{\partial w_{hh}} = \sum_{n=1}^N \frac{\partial L}{\partial w_{hh}(t)} \quad \text{or} \quad \frac{\partial L}{\partial w_{hh}} = \frac{1}{n-1} \sum_{n=1}^N \frac{\partial L}{\partial w_{hh}(t)}$$

כיוון שהמשקלים זהים לאורך כל הרשת, $w_{hh}(t) = w_{hh}$ והשניי בזמן t יהיה רק לאחר ביצוע ה-BPTT והוא רלוונטי רק ליקטור ה- t .

הצורה הפשוטה של ה-BPTT יוצרת בעיה עם הגרדיאנט. נניח שרכיב ה- \hat{y}_i מיוצג באמצעות הפונקציה הבאה:

$$h_t = f(z_t) = f(w_{hh}h_{t-1} + w_{hx}x_t + b_h)$$

לפי כלל השרשרת:

$$\frac{\partial h_n}{\partial x_1} = \frac{\partial h_n}{\partial h_{n-1}} \times \frac{\partial h_{n-1}}{\partial h_{n-2}} \times \dots \times \frac{\partial h_2}{\partial h_1} \times \frac{\partial h_1}{\partial x_1}$$

כיוון ש- w_{hh} קבוע ביחס בזמן עברו וקטו כניסה יחיד, מתקיים:

$$\frac{\partial h_t}{\partial h_{t-1}} = f'(z_t) \cdot w_{hh}$$

אם נציב זאת בכלל השרשרת, נקבל שעבור חישוב הנגזרת $\frac{\partial h_n}{\partial x_1}$ מכפלים $1 - n$ פעמים ב- w_{hh} . לכן אם מתקיים $1 > ||w_{hh}||$ אז הגרדיאנט יתדרדר, ואם $1 < ||w_{hh}||$ הגרדיאנט יתפפס. לעומת זאת, של התדרדרות או התאפסות הגרדיאנט, יכולת להופיע גם ברשותם אחרות, אבל בגלל המבנה של RNN והlienarיות של ה-BPTT בשרותם רקוטטיביות זה קורא כמעט תמיד.

עבור הבעיה של התדרדרות הגרדיאנט ניתן לבצע clipping – אם הגרדיאנט גדול מקבוע מסוים, מנורמלים אותו:

$$\text{if } \|g\| > c, \text{then } g = \frac{cg}{\|g\|}$$

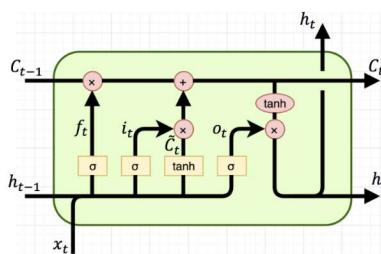
הבעיה של התאפסות הגראדיינט אמונה לא גורמת לחישובים של מספרים עצומים, אך היא בעצם מבלטת את ההשפעה של אברים שמאזים רוחק אחד מהשי. אם למשל יש משפט ארוך, אז במקורה בו הגראדיינט דוער במלר h-TT, BPTT, כמעט והשפעה של המילה הראשונה על המילה الأخيرة. במלים אחרות – התאפסות הגראדיינט גוררת בעיה של Long-term, כלומר קשה ללמידה בעל תלות בטוחו ארוך, כמו משפט ארוך או תופעת שימושנות לאט. בגלל הבעיה הזאת לא משתמשים ב-RNN הקלסי (שנקרא גם Vanilla RNN).

שיפורים, כפי שיאסבירו בפרק הבא.

6.2 RNN Architectures

6.2.1 Long Short-Term Memory (LSTM)

כדי להתגבר על בעיית דעיכת הגראדיינט המונעת מהרשת להשתמש בזיכרון ארוך ווות, ניתן להוסיף אלמנטים לריבב הזיכרון כך שהוא יוכל רק מידע על העבר, אלא יהיה גם בעל שליטה על איך וכמה להשתמש במידע. ב-RNN פשוט לריבב הזיכרון יש שתי כניסה – x_t , h_{t-1} , c_{t-1} , ובעזרתן מחשבים את המוצא על ידי שימוש בפונקציה $LSTM(f_w(h_{t-1}, x_t))$. למעשה ריבב הזיכרון הוא קבוע והמידה מתבצעת רק במשקלים. ב- $LSTM$ יש שני שינויים עיקריים – מלבד הכניסות הרגליות יש עוד כניסה הנכנסת memory cell state ומסומנת ב- c_{t-1} , ובנוסף לכך c_t מחושב בצורה מולכבת יותר. באופן זה האלמנט c_t זורג לזכרון ארוך טווח של דברים, ו- c_t אחראי על הזיכרון של הטווח הקצר. נתבונן בארכיטקטורה של תא הזיכרון:



איור 6.3 תא זיכרון בארכיטקטורת LSTM.

הצמד $[x_t, h_{t-1}]$ נכנס לתא ומוכפל במסקל W , ולאחר מכן עובר בפעולת דורך ארבעה שערים (יש לשים לב שלא מביצים פעולה בין x_t ל- h_{t-1}) הוא שער שכחה והוא אחראי על מחיקת חלק מהזיכרון. אם למשל יש משפט ומוופיע בו נושא חדש, אז שער זה אמרו למחוק את הנושא שהיה שמור בזיכרון. השער השני i_t הוא שער זיכרון והוא אחראי על כמה יש לזכור את המידע החדש לטווח ארוך. אם לדוגמא אכן יש משפט מסוים נושא חדש, אז השער יחליט שיש לזכור את המידע הזה. אם לעומת זאת המידע החדש הוא תיאור שלא רלוונטי להמשך אז אין טעם לזכור אותו. השער השלישי o_t הוא שער שומר מידע על כמה מהמידע רלוונטי לדואטה הנוכחי x_t , כלומר מהי הפלט של התא ביבירן מידע העבר. שיוות השערים האלו קארים מסכות (Masks), וגם מקבלים ערכים בין 0 ל-1 כיוון שהם עוביים דרך סיגמואידי. יש שער נוסף \tilde{c}_t (לפעמים מסומן באות g) שאחראי על השאלה כמה מהזיכרון להזכיר לתא הבא. שער זה משקל את המידע המתkeletal יחד עם i_t שאומר עד כמה יש לזכור להמשך את המידע החדש.

באופן זהה מקבלים חן את h_t שאחראי על זיכרון לטווח הקצר כמו ב-RNN Vanilla, והן את c_t שאחראי על זיכרון של כל העבר. ארכיטקטורת הריבב מאפשרת להתייחס לאלמנטים נוספים נספחים לקשרים לזכרון – ניתן לשכוב חלקיים לא רלוונטיים של התא הקודם (f_t), להתייחס באופן סלקטיבי לכינסה (i_t) ולהציג ריק חלק מהמידע המשקל הקיים (o_t). באופן פורמלי ניתן לנסח את פעולת התא כך:

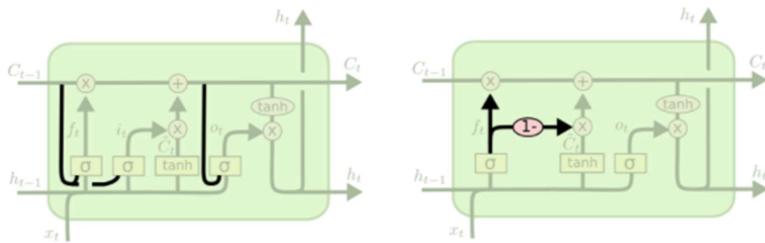
$$\begin{pmatrix} i \\ f \\ o \\ \tilde{c} \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} = \begin{pmatrix} \sigma(w_i \cdot [x_t, h_{t-1}] + b_i) \\ \sigma(w_f \cdot [x_t, h_{t-1}] + b_f) \\ \sigma(w_o \cdot [x_t, h_{t-1}] + b_o) \\ \tanh(w_{\tilde{c}} \cdot [x_t, h_{t-1}] + b_{\tilde{c}}) \end{pmatrix}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, h_t = o_t \odot \tanh(c_t)$$

כאשר האופרטור Θ מסמל כלל איבר איבר (כיוון שלשעים נכנס הזוג $[x_t, h_{t-1}]$, אם נמצא מביצים מफלא מוסמיה, יש לבצע אותה על כל אחד מהאיברים).

יש וריציאות שונות של רכבי LSTM – ניתן למשל לחבר c_{t-1} לא c_t ולקמו h_t אלא גם לשאר השערם. חיבור כזה נקרא peepholes, כיון שהוא מאפשר לשערם להתובן ב- c_{t-1} באופן ישיר. יש ארכיטקטורות שמחברות את c_{t-1} לכל השערם, ויש ארכיטקטורות שמחברות אותו רק לחלק מהשערם. חיבור כל השערם ל- c_{t-1} מוביל את משוואות השערם. במקרה $[x_t, h_{t-1}] + b$, המשוואה החדשה תהיה $(c_{t-1}, x_t, h_{t-1}) + b$.

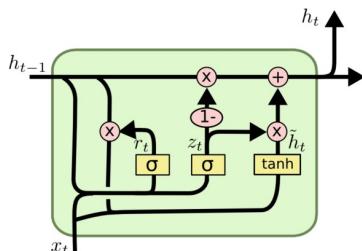
ורויציה אחורית של LSTM מאחדת בין שער השכחה f_t לבין שער החיקון i_t , והחלה עד כמה יש למחוק מידע מהychicon מתקבל ייחד עם החלטה כמה מידע חדש יש לכתוב. שינוי זה ישפיע על המאגר *memory cell*, כאשר במקומם $c_t = f_t \Theta c_{t-1} + (1 - f_t) \Theta \tilde{c}_t$, משווה את העדכון תחתיה: $c_t = f_t \Theta c_{t-1} + i_t \Theta \tilde{c}_t$.



איור 6.4 וריאציות של LSTM – coupled forget and input gates (ימין) ו-peephole connections – (שמאל).

6.2.2 Gated Recurrent Units (GRU)

השנה ארכיטקטורה נוספת של תא זיכרון הנקראת Gated Recurrent Units (GRU), והיא כוללת מספר שינויים ביחס ל-LSTM-4:



אייר 6.5 תא זיכרון בארכיטקטורת GRU.

השיני המשמשות LSTM הוא העובדה שאי memory cell state s_t , וכל השערים מתבססים רק על הקלט והמוחזק של התא הקודם. כדי לאפשר זיכרון הן לטוח ארך והן לטוח קצר, יש שני שערים – Update gate ו-Reset gate – ו�� מחושבים על ידי הנוסחאות הבאות:

Update: $z_t = \sigma(w_z \cdot [x_t, h_{t-1}])$

Reset: $r_t \equiv \sigma(w_r \cdot [x_t, h_{t-1}])$

בازרת שער ה-reset מחשביםCandidate hidden state

$$\tilde{h}_t = \tanh(w \cdot [x_t, r_t \odot h_{t-1}])$$

ראשית יש לשים בפניהם $t \in [0, 1]$ כדי שהוא תוצאה של סיגמאיד.icut נתבונן על \tilde{h} ביחס לרכיב זיכרון פשוט של $f_{w_h h_t}$, כאשר t קרוב ל-1 מתקבל היבויו:Vanilla RNN

$$\tilde{h}_t = \tanh(w \cdot [x_t, r_t \odot h_{t-1}] \approx \tanh(w[x_t, h_{t-1}]) = \tanh(w_{hx}x_t + w_{hh}h_{t-1})$$

המשמעות היא שעבור $1 \rightarrow r_t$ מתקבל רכיב הזיכרון הקלטי, השומר על זיכרון לטוויה קצר. אם לעומת זאת $0 \rightarrow r_t$ אז מתקבל רכיב הזיכרון הקלטי, השומר על זיכרון לטוויה קצר. אם לעומת זאת $0 \rightarrow r_t$ אז מתקבל רכיב הזיכרון הקלטי, השומר על זיכרון לטוויה קצר. אם לעומת זאת $0 \rightarrow r_t$ אז מתקבל רכיב הזיכרון הקלטי, השומר על זיכרון לטוויה קצר.

לאחר החישוב של \tilde{h}_t מחשבים את המצב החבוי בעזרת z_t , שאם הוא מקבל ערכים בין 0 ל-1:

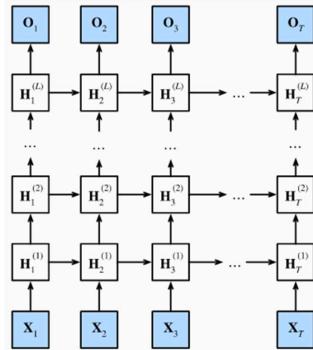
$$h_t = (1 - z_t)\Theta h_{t-1} + z_t\Theta \tilde{h}_t$$

אם $0 \rightarrow 0$, אז $h_t \approx h_{t-1}$, כלומר מעתה מעריכים את המצב הקודם כמו שהוא. אם לעומת זאת $1 \rightarrow 1$ אז $z_t \rightarrow \tilde{h}_t$, כלומר מעריכים מה מצב הקודם כמו שהוא ולקיחים את הערך הנכון. עבור כל ערך אחר של z_t , מקבלים שקול של המצב החבוי הקודם וה- h_t .

ארקיטקטורה זו מאפשרת גם לזכור דברים לאור זמן, וגם מצילה להתחמود עם בעיית הגדריאנט. אם שער העדכון הקרוב ל-1 כל הזמן, אז בעצם מעריכים את המצב החבוי כמו שהוא, ולמעשה הזיכרון נשמר לאור זמן. בנסוף, אין בעיה של הבדורות הגדריאנט, כיון שאם השינוי בין תא לתא לא גדול, אז הגדריאנט קרבע ל-1 כל הזמן ולא מתבודר.

6.2.3 Deep RNN

עד כה דובר על רכיבי זיכרון ייחודיים, שנitin לשדרר אותם יחד ולקבל שכבה שיכולה לנתח. ניתן להרחיב את המודל הפשוט לרשות בעל מספר שכבות עמוקות.



.איור 6.6 ארכיטקטורת Deep RNN.

נתאר את הרשת באופן פורמלי. בכל נקודת זמן t יש וקטור כניסה $x_t \in \mathbb{R}^{n \times d}$ (וקטור בעל n איברים, כאשר כל איבר הוא מממד d). איברי הסדרה נכנים לרשת בעלת L שכבות $-T$ איברים בכניסה, אשר עבר כל נקודת זמן t יש L שכבות (או מוצבים חビויים). כל שכבה מכילה h מוצבים חビויים, כאשר השכבה ה- ℓ בנקודת זמן t מוסמנת בתווך $H_t^{(\ell)} \in \mathbb{R}^{n \times h}$. ככל נקודת זמן t יש גם וקטור מוצוא באורך q – $o_t \in \mathbb{R}^{n \times q}$. נסמן: $x_t = H_t^{(0)}$, ונניח שבין שכבה אחת לשנייה משתמשים באקטיבציה לא ליניארית ϕ . בעזרת סימונים אלה נקבל את הנוסחה הבאה:

$$H_t^{(\ell)} = \phi_\ell \left(H_t^{(\ell-1)} w_{xh}^{(\ell)} + H_{t-1}^{(\ell)} w_{hh}^{(\ell)} + b_h^{(\ell)} \right)$$

כאשר $w_{xh}^{(\ell)} \in \mathbb{R}^{h \times n}$, $w_{hh}^{(\ell)} \in \mathbb{R}^{h \times h}$, $b_h^{(\ell)} \in \mathbb{R}^{1 \times h}$ הפלט o_t תלוי באוף ישיר רק בשכבה ה- L , והוא מחושב על ידי:

$$o_t = H_t^{(L)} w_{hq} + b_q^{(L)}$$

כאשר $w_{hq}^{(L)} \in \mathbb{R}^{h \times q}$, $b_q^{(L)} \in \mathbb{R}^{1 \times q}$ הם הפרמטרים של שכבת הפלט.

ניתן כמובן להשתמש במצבים החבויים ברכיבי זיכרון LSTM, וכך לקבל GRU או LSTM.

6.2.4 Bidirectional RNN

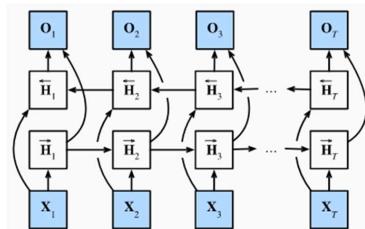
כל הרכיבים והרשאות שנידונו עד כה עוסקו בסדרות סיבתיות, כלומר, סדרות בהן כל איבר מושפע מ קודמי או מאלו הבאים אחריו. למשל, ערך מסוימת ביום מסוימת קשור לערכה ביום הקודמים, אך הערך שליה ביום המחרת (שכלל עוד לא-'ידע) לא מושפע בשום צורה על ערכיה ביום הנוכחי. דוגמא נוספת – התפוחות של גל בזמן תלייה בערבי הקודמים של הגל אך אינה מושפעת ממצבי הגל בעטדי. זהו אמן מצב היותר מצוין, אך ישם מבנים בהם יש סדרה לאו דווקא סיבתית, כך שניתן לבחון את הקשר בין איברייה מסוימים הקיימים. ניקח לדוגמה את משימת ההשלמה הבאה:

I am _.

I am _ hungry.

I am _ hungry, and I can eat a big meal.

כעת נניח שבכל אחד מהמשפטים צריך לבחור את אחת מהמילים שבסט $\{ \text{happy, not, very} \}$. מבון סוף הביטוי, במקורה וק"י, תורם מידע שימושי על איזו מילה לבחור. מודול שאינו מסוגל לנצל את המידע לאחר המילה החסורה יכול לפחות מילוי חשוב, ורובה יכול לנחש מיליה שאינה מסתדרת עם המשך המשפט זו מבחן תחבירית והן מבחינת המשמעות. כדי לבנות מודל שמתאים לכך כל חלקי המשפט, יש לתוכנן ארכיטקטורה שמאפשרת לנתח סדרה מסוימת היעיונים שלה. ארכיטקטורה כזו נקראת RNN דו-방ת, Bidirectional RNN, והוא למעשה פעולה מוצעת יתוח של סדרה מסוימת היעיונים שלה במקביל. באופן זהה כל מצב חבוי נקבע בו זמני על ידי הנתונים של שני מצבים חבויים אחרים – זה שלפני וזה לאחריו.



איור 6.6 ארכיטקטורת RNN דו-방ת

עבור כל כניסה $x_t \in \mathbb{R}^{n \times d}$ נחשב במקביל שני מצבים חבויים – $\vec{H}_t \in \mathbb{R}^{n \times h}$, $\tilde{H}_t \in \mathbb{R}^{n \times h}$, כאשר h זה מספר רכיבי ההזיכרון בכל מצב חבוי. כל מצב מחושב באופן הבא:

$$\vec{H}_t = \phi(x_t w_{xh}^{(f)} + \vec{H}_{t-1} w_{hh}^{(f)} + b_h^{(f)})$$

$$\tilde{H}_t = \phi(x_t w_{xh}^{(b)} + \tilde{H}_{t+1} w_{hh}^{(b)} + b_h^{(b)})$$

כאשר $w_{xh}^{(b)} \in \mathbb{R}^{d \times h}$, $w_{hh}^{(b)} \in \mathbb{R}^{h \times h}$, $b_h^{(b)} \in \mathbb{R}^{1 \times h-1}$ $w_{xh}^{(f)} \in \mathbb{R}^{d \times h}$, $w_{hh}^{(f)} \in \mathbb{R}^{h \times h}$, $b_h^{(f)} \in \mathbb{R}^{1 \times h}$ הפרמטרים של המודל. לאחר החישוב של \vec{H}_t ו- \tilde{H}_t מושרים אותם יחד ומתקבלים את $H_t \in \mathbb{R}^{n \times 2h}$, ובעזרתו מחשבים את המוצא:

$$o_t = H_t w_{hq} + b_q$$

כאשר $w_{hq} \in \mathbb{R}^{2h \times q}$, $b_q \in \mathbb{R}^{1 \times q}$.

6.2.5 Sequence to Sequence Learning

ב

<https://buomsoo-kim.github.io/blog/tags/#attention-mechanism>

6. References

Vanilla:

<http://karpathy.github.io/2015/05/21/rnn-effectiveness/>

LSTM, GRU:

<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Deep RNN, Bidirectional RNN:

http://d2l.ai/chapter_recurrent-modern/index.html

7. Deep Generative Models

המודלים שהוצעו בפרקם הקודמים הינם מודלים דיסקרטיניביים, קרי הם מאמנים לבצע פעולות על בסיס נתונים, אך לא יכולים ליצור פיסות מידע או דוגמאות חדשות בעצמן. בינו לבין אליהם קיימים מודלים גנרטיביים, המסוגלים ליצור פיסות מידע חדשות על בסיס הדוגמאות שלמנדו. באופן פורמלי, בהינתן אוסף דוגמאות $\mathcal{X} \in \mathbb{R}^{n \times d}$ ו敖פ' $\mathbb{R}^d \in \mathcal{Z}$, מודל דיסקרטיבי מאשר לשרוך את ההסתברות $(\mathcal{Z}|x)$. מודל גנרטיבי לעומת זאת לומד את ההסתברות $(x|\mathcal{Z})$ במקורה שהתగיות אין נתונים), כאשר x , הן צמד נתון של דוגמא - label, מתוכן ניתן לייצר דוגמאות חדשות.

ישנם שני סוגים עיקריים של מודלים גנרטיביים: סוג אחד של מודלים ממואן למצואו באופן מפורש את פונקציית הפילוג של הדאטה הנתנו, ובעזרת הפילוג לייצר דוגמאות חדשות (על ידי דגימה מההתפלגות שלמנדה). סוג שני של מודלים אינם עוסקים בשערוך הפילוג של הדאטה המקורי, אלא מסוגל לייצר דוגמאות חדשות בדרךים אחרות. בפרק זה נדון במודלים הפופולריים בתחום – VAE, GANs, והשיכים לשוג הריאשן והשי של המודלים האנרטיביים.

7.1 Variational AutoEncoder (VAE)

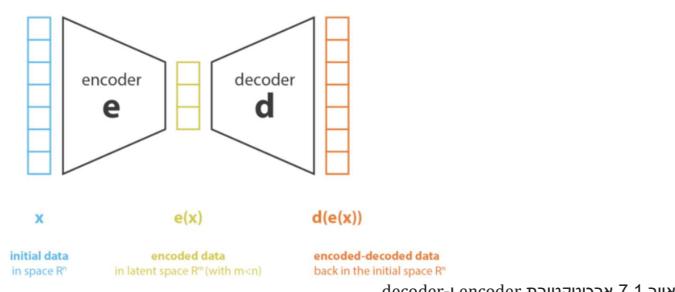
המודל הראשון הינו VAE, וכדי להבין כיצד ניתן בעזרתו לייצר מידע חדש, יש להסביר קצת מהם Autoencoders. Autoencoder הינו רשות המאפשרת לבצע הורדת מידע, תוך איבוד מינימלי של מידע. בכדי להבין את המבנה ואופי הפעולה שלו, נקדים 먼저 על הורדת מידע באופן כללי.

7.1.1 Dimensionality Reduction

במקרים רבים, הדאטה אותו רוצים לנתח הוא בעל ממד גבוה, למשל, כל דגימה יש מספר רב של מאפיינים (features). לרוב, לא כל המאפיינים משמעותיים באותה מידה. לדוגמה – מחיר מנעה של חברה מסוימת מושפע ממספר רב של גורמים, אך ככל הנראה גובה ההכנסות של החברה יותר מאשר הגיל הממוצע של העובדים. דוגמא נוספת – במשמעות חזוי גיל של אדם על פי תמנות פנים שלו, לא כל הפיסולים בתמונה פנים יהיו בעלי חשיבות לצורך החיזוי. כיוון שהקשה לנתח דאטה גבוהה יחסית לפיסולים עבור דאטה כזו, במקרים רבים מנסה להוריד את הממד של הדאטה תוך איבוד מידע מינימלי עד כמה שניתן. בהתאם להורדת הממד מנסים לקבל ייצוג חדש של הדאטה בעל ממד יותר נמוך, כאשר הייצוג הזה מורכב מהמאפיינים הנמורים יותר, בו באים לידי ביטוי רק המאפיינים המשמעותיים של הדאטה.

הייצוג של הדאטה בממד נמוך נקרא הייצוג הלטנטי (חבי) או הקוד הלטנטי, ומאוור, יותר קל לבנות מודלים למשימות שונות על סמך הייצוג הלטנטי של הדאטה מאשר לעבד עם הדאטה המקורי. בכדי לקבל ייצוג הלטנטי איזוטטי, ניתן לאמן אותו באמצעות מנגנון הנקרא decoder, הבוחן את יכולת השחזור של הדאטה מהייצוג הלטנטי שלו. ככל שניתן לשחזור בצהורה מדויקת יותר את הדאטה המקורי, ככלומר אובדן המידע בתהילך הוא קטן יותר, כך הקוד הלטנטי אכן מייצג בצהורה אמינה את הדאטה המקורי.

תהליך האימון הוא דו שלבי: פיסות מידע המייצגת על ידי encoder המאפיינים $\mathcal{X} \in \mathbb{R}^n$ עובר דרך encoder, שמרתנו להפוך מהדאטה את הייצוג הלטנטי שלו $\mathcal{Z} \in \mathbb{R}^m$ ($\mathcal{Z} = e(x)$, כאשר $n > m$). לאחר מכן התוצאה מוכנסת לdecoder-encoder..decoder לשחזור את הדאטה המקורי, ולבסוף מתקבל וקטור $\mathcal{X}' \in \mathbb{R}^n$ ($\mathcal{X}' = d(e(x))$. $d(x) = d(e(x))$. $e(x) = x$) אס מתקיים השוויון $d(e(x)) = x$ אז למעשה לא אבד שום מידע בתהילך, אך אם לעומת זאת ($\mathcal{Z} = d(e(x))$, $\mathcal{Z} \neq x$) אז מידע מסוים אבד עקב הורדת הממד ולא היה ניתן לשחרר אותו במלואה בפונחו. באופן אינטואטיבי, אם אנו מצילים לשחרר את הדאטה המקורי מיצוגו של ממד הנמוך בבדיקה טוב מספיק, נראה שאהיצוג הלטנטי הצליח להפיך את המאפיינים המשמעותיים של הדאטה המקורי.

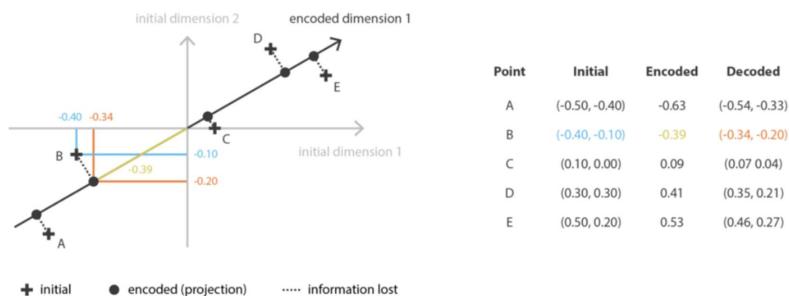


כאמור, המטרה העיקרית של השיטות להורדת ממד הינה לקבל ייצוג לטני אינטלי עד כמה שניתן. הדריך לעשوت זאת היא לאמן את זוג encoder-decoder השומרים על מקסימום מיעע בעת הקידוד, ובאים לミニימים את שגיאות שחזור בעת הפענוח. אם נסמן בהתאם E ו-D את כל הזוגות של encoder-decoder האפשריים, ניתן לנסח את בעיית הורדת הממד באופן הבא:

$$(e^*, d^*) = \arg \min_{(e,d) \in E \times D} \epsilon(x, d(e(x)))$$

כאשר $\epsilon(x, d(e(x)))$ הוא שגיאת השחזור שבין הדטה המקורי לבין הדטה המשוחזר.

אחד השימושות העיקריים להורדת ממד שאפשר להסתכל עליו בצורה PCA (Principal Components Analysis). בשיטה זו מטילים (בצורה ליניארית) דadata מממד n לממד $m < n$, כך שהמאפיינים של הייצוג הלטנטי של הדוגמאות המקוריות יהיו אורתוגונליים. תהליך זה נקרא גם *feature decorrelation*, והמטרה שלו היא למצער את המרכיב האקלידי בין הדטה המקורי לדאטה המשוחזר, בצורה ליניארית גם כן, מהויצוג החדש במרחב m -ממדי.

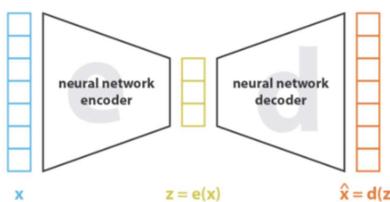


איור 7.2 דוגמא להורדת ממד בשיטת PCA.

במושגים של encoder-decoder, ניתן להראות כי הזוג PCA מփש את הזוג של encoder-decoder שמקיימים שני תנאים: א. ה-encoder מבצע טרנספורמציה ליניארית על הדטה כך שהמאפיינים החדשים (בממד נמוך) של הדטה יהיו אורתוגונליים. ב. ה-decoder הליניארי המתאים יביא לשגיאה מינימלית במונחים של מרחק אקלידי בין הדטה המקורי לבין הדטה המשוחזר מהויצוג החדש. ניתן להוכיח שה-encoder האופטימי מכיל את הוקטוריים העצמיים של מטריצת covariance של מטריצת $\text{cov}(x)$, וה-decoder הוא השולוף של encoder-design.

7.1.2 Autoencoders (AE)

ניתן לקחת את המבנה של ה-encoder-decoder המתוואר בפרק הקודם ולהשתמש בראשת נירונים עבור בניתן הייצוג החדש ועבור השחזור. מבנה זה נקרא Autoencoder:



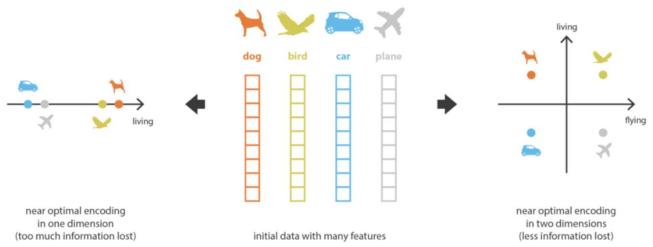
$$\text{loss} = \|x - \hat{x}\|^2 = \|x - d(e(x))\|^2$$

איור 7.3 – שימוש בראשות נירונים עבור הורדת הממד והשחזור.

באופן זהה, הארכיטקטורה יוצרת לדטה צואר בקבוק מידע (information bottleneck), שמבטיח שרק המאפיינים החשובים של הדטה, שבאמצעותם ניתן לשחזר אותה בדיוק טוב, ישמשו לייצוג במרחב הלטנטי. במקרה פשוט בו בבל רשות שיש רק שכבה חכיה אחת והוא לא משתמש בפוקנציות הפעלה (activation functions) לא ליניאריות, ניתן לראות כי ה-encoder יחשוף טרנספורמציה ליניארית של הדטה, שבאמצעותו ניתן לשחזר

באופן לינארי גם כן. בדומה ל-PCA, גם רשת כזו תחשוף להויריד את הממד באמצעות טרנספורמציות לינאריות של המאפיינים המקוריים אך היצוג בממד נמוך המפק על ידה לא יהיה בהכרח זהה של PCA, כמו שלhalbידל מ-PCA המאפיינים החדשניים (לאחר הודת ממד) עשויים לצאת לא אורתוגונליים (קורולציה שונה מ-0).

כעת נניח שהרשנות של ה-encoder וה-decoder הן רשות עוקחות ומשתמשות בפונקציות הפעלה לא לינאריות. במקרה זה, ככל שאררכיטקטורה של הרשותות מורכבת יותר, רשת ה-encoder יכולה להויריד יותר ממדים תוך יכולת באמצעות decoder-הויריד לבעצם שזרז לאן לא איבוד מידע. באופן תיאוטרי, אם ל-encoder-ויל-decoder יש מסווק דרגות חופש (למשל מסווק שכבות ברשת נירונית), ניתן להפחית ממד של>Data לא כל איבוד מידע. עם זאת, הפחיתת ממד דרישות שכך עלולה לגרום לדעתה המשוחזר לאבד את המבנה שלו. לכן יש חשיבות גדולה בברירות הממדים של המרכיב הלטני, כך שמצד אחד ניתן יתבצע ניפוי של מאפיינים פחות משמעותיים ומצד שני המידע עדיין יהיה בעל שימושים שונים. כדי להמחיש את המתואר לעיל,ijk להדגמא מערכת שמקבלת תמונות של כלב, ציפור, מכונית ומטוס ומנסה למצוא את הפרמטרים העיקריים המבוחנים ביניהם:



איור 7.4 דוגמא לשימוש ב-encoder-decoder.

לפרטים אלו יש הרבה מאפיינים, וקשה לבנות מודל שمبחר בויהם על סמך כל המאפיינים. רשת נירונים מורכבת מסווק מאפשרת לבנות ייצוג של כל הדוגמאות על קו ישר, כך שכל שפרט מסוים נמצא יותר ימין, כך הוא יותר "חי". באופן זה אגסם מתקבל ייצוג חד-ממדי, אבל הוא גורם לאיבוד המבנה הסטטי של הדוגמאות ולא באמת ניתן להבini את ההפרדה ביניהם. לעומת זאת ניתן להויריד את הממד של תמונות אלו לדוו-ממד ולהתיחס רק לפתרומים "חי" ו"עף", וכך לקבל הבחנה יותר ברורה בין הדוגמאות. כמובן שה הפרדה זו היא הרבה יותר פשוטה מאשר הסתככות על כל הפרמטרים (הפיזסלים) של הדוגמאות. דוגמא זו מראה את החשיבות שיש בבחירה הממדים של ה-encoder.

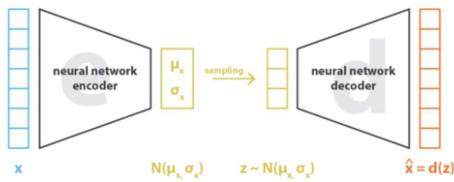
7.1.3 Variational AutoEncoders (VAE)

ניתן לקחת את ה-AE ולהפוך אותו למודל גנרטיבי, כלומר מודל שמסוגל ליצור בעצמו דוגמאות חדשות שאכן מתפלגות כמו הפליג של הדאטה המקורי. אם מזכיר בזומאי של תמונות למשל, אז רצתה שהמודול יהיה מסוגל ליצור תמונות שנראות אוננטיות בחוץ לדאטה עליי אומן. AE מאמין לייצג את הדאטה בממד נמוך, שליקוב בשבעון התמאפיינים העיקריים, ולאחר מכן משוחרר את היציאה לממד המקורי. אולם, מנגנון זה אינו מסוגל להשליך עליון בו הדאטה מייצג במרחב הלטני. אם יוגרך וקטור לשמהו מהמרחב הלטני – קרוב לוודאי שהוא ייזוג שdomה לדאטה המקורי. אם היינו מכנים אותו ל-decoder, סביר להניח שההתוצאה לא תהיה דומה בכלל לדאטה המקורי. למשל אם AE אומן על אוסף של תמונות של כלבים ודוגמאות באקראי וקטור מהמרחב הלטני שלו, הסתמי קיבל תמונת כלב כלשהו לאחר השזרתו של ה-decoder.

כדי להימנע עם בעיה זו, ניתן להשתמש ב-VAE. בשונה מ-AE שלוקח דאטה ורק בונה לו ייצוג מממד נמוך, קובע התפלגות פריריות למרחב הלטני z – למשל התפלגות נורמלית עם תוחלת 0 ומוריצ'זת covariance Σ . בהינתן התפלגות זו, רשת ה-encoder מאמנת ללקב דאטה x ולהציג פרמטרים של התפלגות פוטנציאלית $p_{\text{latent}}(z|x)$, מתוך מטרלה כמה שנטען את המרחק בין התפלגות z ו- $x|z$. ואחר מכן מעתיקים מההתפלגות הפוטנציאלית $p_{\text{latent}}(z|x)$ (הנתונה על ידי הפרמטרים המוחובים ב-encoder), ומעבירים אותם דרך decoder כדי לייצר פרמטרים של התפלגות $p_{\text{latent}}(x|z)$. חשוב להבהיר שגם הדאטה המקורי הוא תמונה המורכבת ממאסף של פיקסלים, אז בזומאי יקבל $z|x$ לכל פיקסל בוגרפדי ומההתפלגות רצוי דוגמאות נקודה שתציג את ערך הפיקסל בתמונה המשוחזרת.

באופן זהה, הלמידה דואגת לא רק להורדת הממד של הדטה, אלא גם להתפלגות המשורית על המרחב הlatent. כאשר התפלגות המונטנית בmoza $z|x$ טוביה, קרי קרובה לתפלגות המקורית של x , ניתן בעדרתה גם ליצור דוגמאות חדשות, ובצム מתקבל מודל גנרטיבי.

כאמור, ה-VAE מנסה לייצג את הדטה המקורי באמצעות התפלגות בממד נמוך יותר, למשל התפלגות נורמלית עם תוחלת וטראיצית covariance: $N(\mu_x, \sigma_x) = N(z|x)$. z – השוב לשיסם לב להבדל בתפקידו של ה-decoder – بعد ש-VAE הוא נועד לתחילה האימון בלבד ובפועל מה שחשב זה הייצוג הלטני, ה-VAE-Cheshkov שוב לא פחות מאשר הייצוג הלטני, כיון שהוא זה שמשמש ליצירת DATA חדשה לאחר תהליכי האימון, או במקרים אחרות, הוא הופך את המערכת למודול גנרטיבי.



אייר 7.5 ארכיטקטורה של VAE.

לאחר שהציג המבנה הכללי של VAE, ניתן לתאר את תהליכי האימון, ולשם כך נפריד בשלב זה בין שני החלקים של ה-VAE.encoder מאמן רשת שמקבלת דוגמאות מסט האימון, ומנסה להפיק ממנה פרמטרים של התפלגות $z|x$ הקוראים כמה שפיטן לפרמטרים של התפלגות הפרויוית, z , שcamor נקבעה מראש. מלהתפלגות הננדמת זו דוגמים וקטורים לטנטים חדשים ומעברים אותם ל-decoder. ה-decoder מבצע את הפעולה הפוכה – לוחץ וקטר שגדם מהמרחב הלטני $z|x$, ומיציר באמצעותו דוגמא חדשה שאמורה להיות דומה לדומה לדומה לאיטה המקורי. תהליכי האימון יהיה זהה שמייצר את השגאה של שני חלקים VAE – גם $z|x$ שבוצוא ייה כמו שיוצר קרוב ל- x המקורי, וגם התפלגות $z|x$ תהיה כמו שיוצר קרובות לתפלגות הפרויוית, z .

נתאר באופן פורמלי את בעיית האופטימיזציה של VAE. נסמן את הוקוטרים של המרחב הלטני ב- z , את הפרמטרים של ה-decoder ב- θ , ואת הפרמטרים של encoder ב- λ . כדי למצוא את הפרמטרים האופטימליים של שתי הרשותות, נרצה להביא למינימום את $(\theta; \lambda)$, כלומר למצוא את הנראות המרבית של סט האימון תחת θ . כיון שפונקציית $\log p$ מונוטונית, ככל לחתך את לוג ההסתברות:

$$L(\theta) = \log p(x; \theta)$$

אם בביא למינימום את הביטוי הזה, נקבל את θ -האופטימי. כיון שלא ניתן לחשב במפורש את $(\theta; \lambda)$, יש להשתמש בקירוב. נניח והפלט של encoder הוא בעל התפלגות $q(z|\lambda)$ (מה ההסתברות לקבל את z בהינתן λ בכינסה), וננסה לייצג התפלגות זו באמצעות רשת נירונים עם סט פרמטרים λ .icut ניתן לוחץ ולהכפיל את $:q(z; \lambda) L(\theta)$

$$\log p(x; \theta) = \log \sum_z p(x, z; \theta) = \log \sum_z q(z; \lambda) \frac{p(x, z; \theta)}{q(z; \lambda)} \geq \sum_z q(z; \lambda) \log \frac{p(x, z; \theta)}{q(z; \lambda)}$$

כאשר אי השווין האחרון נבע [mai-shoivin yotam](#), והביטוי שמיין לאי השווין נקרא Evidence Lower Bound (ELBO(θ, λ)). ניתן להוכיח שההפרש בין ה-ELBO(θ, λ) לבין הערך שלפני הקירוב הוא המרחק בין שתי התפלגות $:D_{KL}(q(z|x), p(z|x))$, שנקרא Kullback-Leibler divergence ומסומן ב-

$$\log p(x; \theta) = ELBO(\theta, \lambda) + D_{KL}(q(z; \lambda) || p(z|x; \theta))$$

אם שתי התפלגות זהות, אז מערך D_{KL} ביןיהם הוא 0 ומתקבל שוויון: $\log p(x; \theta) = ELBO(\theta, \lambda)$. כזכור, אנחנו מחפשים למינימום פונקציית המבחן $k(x; \theta)$, וcut בעדרת הקירוב ניתן לרשום:

$$L(\theta) = \log k(x; \theta) \geq ELBO(\theta, \lambda)$$

$$\rightarrow \theta_{ML} = \arg \max_{\theta} L(\theta) = \arg \max_{\theta} \max_{\lambda} ELBO(\theta, \lambda)$$

cut ניתן בעדרת שיטת Gradient Descent (GD) למציאת האופטימום של הביטוי, וממנו להפיק את הפרמטרים האופטימליים של ה-VAE. נפתח יותר את ה-ELBO(θ, λ) עבור encoder וה-decoder, ביחס לשתי התפלגות:

$p(x|z; \theta)$ – ההסתברות ש-*decoder*-*encoder* עם סט פרמטרים θ יוציא x בהינתן z .
 $q(z|x; \lambda)$ – ההסתברות ש-*encoder* עם סט פרמטרים λ יוציא את z בהינתן x בכניסה.

לפי הגדרה:

$$\begin{aligned} ELBO(\theta, \lambda) &= \sum_z q(z|x; \lambda) \log p(x, z; \theta) - \sum_z q(z|x; \lambda) \log q(z|x; \lambda) \\ &\quad : p(x, z) = p(x|z) \cdot p(z) \text{ ניתן לפתחו לפי חוק ב'יס } p(z) \cdot p(x|z) \\ &= \sum_z q(z|x; \lambda) (\log p(x|z; \theta) + \log p(z; \theta)) - \sum_z q(z|x; \lambda) \log q(z|x; \lambda) \\ &= \sum_z q(z|x; \lambda) \log p(x|z; \theta) - \sum_z q(z|x; \lambda) (\log q(z|x; \lambda) - \log p(z; \theta)) \\ &= \sum_z q(z|x; \lambda) \log p(x|z; \theta) - \sum_z q(z|x; \lambda) \frac{\log q(z|x; \lambda)}{\log p(z; \theta)} \end{aligned}$$

הביטוי השני לפי הגדרה שווה ל- $\mathcal{D}_{KL}(q(z|x; \lambda) \| p(z; \theta))$, כלומר:

$$= \sum_z q(z|x; \lambda) \log p(x|z; \theta) - \mathcal{D}_{KL}(q(z|x; \lambda) \| p(z))$$

הביטוי הראשון הוא בדיק התוחלת של $\log p(x|z; \theta)$. תחת ההנחה ש- z מתייחס לנורמלית, ניתן לרשום:

$$= \mathbb{E}_{q(z|x; \lambda)} \log N(x; \mu_\theta(z), \sigma_\theta(z)) - \mathcal{D}_{KL}(N(\mu_\lambda(x), \sigma_\lambda(x)) \| N(0, I))$$

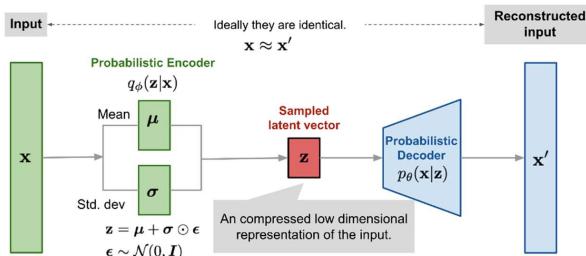
כדי לחשב את התוחלת ניתן פשטוט לדגם דוגמאות מההתפלגות $(\mu_\theta(x), \sigma_\theta(x))$ ולקבל:

$$\mathbb{E}_{q(z|x; \lambda)} \log N(x; \mu_\theta(z), \sigma_\theta(z)) \approx \log N(x; \mu_\theta(z), \sigma_\theta(z))$$

עבורו הביטוי השני יש נוסחה סגורה:

$$\mathcal{D}_{KL}(N(\mu, \sigma^2) \| N(0, I)) = \frac{1}{2} (\mu^2 + \sigma^2 - \log \sigma^2)$$

icut משיש בידינו נוסחה לחישוב פונקציית המחיה, נוכל לבצע את ההליך הלמידה. יש לנו לב שפונקציית המחיה המקורית הייתה מילוי רק ב- θ , אך באופן שפויהנו אותה היא למעשיה דואגת גם למינור הפרש בין הכניסה של encoder לבין המוצא שלו, וגם למינור המרחק בין ההתפלגות הפריאוית של z לבין ההתפלגות $x|z$ שבמוצאת *VAE*.



$$\begin{aligned} x_t &\rightarrow \mu_\lambda(x_t), \Sigma_\lambda(x_t) \rightarrow z_t \sim \mathcal{N}(\mu_\lambda(x_t), \Sigma_\lambda(x_t)) \rightarrow \mu_\theta(z_t), \Sigma_\theta(z_t) \\ \text{ELBO} &= \sum_t \log \mathcal{N}(x_t; \mu_\theta(z_t), \Sigma_\theta(z_t)) - \mathcal{D}_{KL}(\mathcal{N}(\mu_\lambda(x_t), \Sigma_\lambda(x_t)) \| \mathcal{N}(0, I)) \end{aligned}$$

אייר 7.6 תהליכי הלמידה של VAE.

כאשר נתון אוסף דוגמאות X , ניתן להעביר כל דוגמא x כ- encoder ולקבל עבורה את μ_x, σ_x . לאחר מכן נקבע וקיים לטוני z מהתפלגות עם פרמטרים אלו, מעבירים אותו כ- decoder ומקבלים את $\sigma_\theta, \mu_\theta$. לאחר התהיליך ניתן להציג את הפרמטרים המתפללים ב-ELBO-0 ולחשב את ערך פונקציית המחר. ניתן לשים לב שה-ELBO-0 מושך משני – האיבר הראשון מושך את הדמיון בין הדוגמא שבעיסיה לבין התפלגות שמתקבלת במצב, והאיבר השני מבצע רגולרייזציה להתפלגות הפriorית במרחב הלטנטי. הרגולרייזציה גורמת לכך שההתפלגות במרחב הלטנטי z תהיה קרובה עד כמה שניתן להתפלגות הפriorית z . אם ההתפלגות במרחב הלטנטי קרובה להתפלגות הפriorית, אז ניתן באמצעות decoder ליצור דוגמאות חדשות, ובמובן זה ה-VAE הוא מודל גנרטיבי.

הdagima של z מהתפלגות במרחב הלטנטי יוצרת קשיי בחישוב האגדיאנט של ה-ELBO-0, אך בדרך כלל מוצע Reparameterization trick – דוגמאות z_0 מהתפלגות נורמלית סטנדרטית, ואנו כדי לקבל את ערך הדגימה של z משתמשים בפרמטרים של ה- encoder : $(x) = \mu_0 + \sigma_0 \lambda(x)$. בגישה זו כל התהיליך יהיה דטרמיניסטי – מגרלים z_0 מראש ואך רק נשאר לחשב באופן סכמטי את התפישות הערך ברשות.

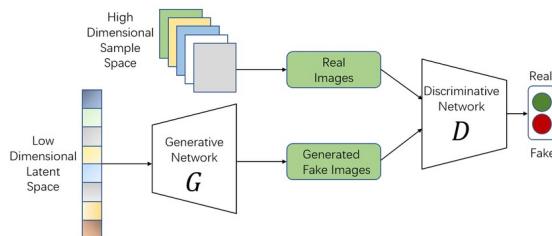
7.2 Generative Adversarial Networks (GANs)

GANs או Generative Adversarial Networks, ובשונה מ-VAE, גישה אחרת של מודל גנרטיבי נקראת Generative Adversarial Networks. גישה זו לא מנוסמת לשערר התפלגות של DATA בקרה מפוזרת (על ידי מציאת הפרמטרים המתקיימים בתפלגות DATA), אלא יוצרים DATA לאמן שטי רשותה במקביל – רשות אחת שלומדת לייצר דוגמאות, ורשות שנייה שלומדת לבחין בין תמונה סינטטית שנוצרה על ידי הרשות הראשונה. הרשות הראשונה מאומנת ליצור דוגמאות שיגרמו לרשות השנייה לחשב שהן אמיתיות, בזמן שהמתירה של הרשות השנייה היא לא לתת לרשות הראשונה לבלב אותה. באופן זה הרשות הראשונה מהווה למעשה מודל גנרטיבי, בעוד שלב האימון היא מסוגלת לייצר DATA סינטטי שלא ניתן להבחין בין DATA אמיתי.

7.2.1 Generator and Discriminator

בפרק זה נסביר את המבנה של ה-GAN הקלאסי שהומצא בשנת 2014 על ידי Ian Goodfellow ושותפיו. נזכיר כי קיימים מאות רבות של וריאציות שונות של GAN שהוצעו מאז, ועדין בתחום זה פועל מאוד מחקריה.

כאמור, GAN מבוסס על שני אלמנטים מרכזיים – רשות שיזירת DATA (generator) ורשות שמכרעה האם DATA ה- D (discriminator). כאשר האימון נעשה על שתי הרשות יחד. ה- G - D discriminator מקבל כקלט DATA אמיתית או סינטטית (discriminator), כדי למדוד לבחין בין DATA אמיתית לבין DATA סינטטי. ה- G - D generator מיציר DATA ומתקבל פידבק מה- D generator- D דוגמאות שנראות אמיתיות. בסמן את ה- G -ב- G ואת D - D , וכן את הסכמה הבאה:



איור 7.7 ארכיטקטורת GAN.

D discriminator הוא למעשה מושג שהוא מושג שהפלט שלו הוא ההסתברות שהקלט הינו DATA אמיתי, ונסמן ב- $D(x)$ את ההסתברות ה- x . כדי לאמן את ה- G - D discriminator נרצה שני דברים: א. למקסם את $D(x)$ עבור x מסוים האימון, כמובן, לטעות כמה שפחות בזיהוי DATA אמיתי. ב. לחזער את $D(x)$ עבור DATA סינטטי, כמובן, להחות כמה שיוצר DATA תhypה כמה שייצור DATA על ידי ה- G - D . במקרה דומה נרצה לאמן את ה- G - D generator שנדגימות שהוא מייצר DATA כמו שיוצר DATA דומות לDATA האמיתיים, כמובן גורם לכך generator- D מונען לגורם לו שבייל לאמן ייחד את שני חלקים המודל, בניית פונקציית מבחן בעלת שני איברים, באופן הבא:

$$V(D, G) = \min_G \max_D \mathbb{E}_{x \sim \text{Data}} \log D(x) + \mathbb{E}_{z \sim \text{Noise}} \log (1 - D(G(z)))$$

נסביר את הביטוי המתkeletal: ה- z -generator מעוניין למקסם את פונקציית המחר, כך שהערך של $D(G(z))$ יהיה כמו שיתור קרוב ל-1 ו- $D(G(z))$ יהיה כמו שיתור קרוב ל-0. ה- z -generator לעומת זאת רוצה להביא למינימום את פונקציית המחר, וכך $D(G(z))$ יהיה כמו שיתור קרוב ל-1, כלומר ה- z -generator יחשב $S(z)$ הוא דатаה אמיתית.

כעת האימון נעשה באופן איטרטיבי, כאשר פעם אחת מקבעים את G ומאמנים את D , ופעם אחרת מקבעים את D ומאמנים את G . אם מקבעים את G , אז למעשה מאמנים מסווג בינהר, כאשר מוחפשים את האופטימום התליי בוקטור הפרמטרים ϕ_d :

$$\max_{\phi_d} \mathbb{E}_{x \sim Data} \log D_{\phi_d}(x) + \mathbb{E}_{z \sim Noise} \log \left(1 - D_{\phi_d}(G_{\theta_g}(z)) \right)$$

אם לעומת זאת מקבעים את D , אז ניתן להתעלם מהאיבר הראשון כיוון שהוא פונקציה של ϕ_d בלבד וקבוע ביחס ל- θ_g . לכן נשאר רק לבדוק את הביטוי השני, שמחפש את ה- z -generator שמייצר דатаה שנאה אמיתית בזורה הטובה ביותר:

$$\min_{\theta_g} \mathbb{E}_{z \sim Noise} \log \left(1 - D_{\phi_d}(G_{\theta_g}(z)) \right)$$

כאמור, המטרה היא לאימן את G בעזרת D (במצביו המקורי), כדי שהיא מסוגל ליצור דוגמאות הנראות אוטנטיות. האימון של ה- z -generator נעשה באמצעות Gradient Descent (מוצע פונקציית המחר ביחס ל- θ_g), והאימון של ה- D -discriminator נעשה באמצעות Gradient Ascent (מוצע פונקציית המחר ל- ϕ_d). האימון מתבצע בשני מספר מסויים של Epochs, כאשר כאמור מאמנים ליטירגון את G ו- D . בפועל דוגמים mini-batch בגודל m מסט האימון (x_m, \dots, x_1) ו- m דוגמאות של רעש (z_1, \dots, z_m), ומכניםים את הקלט G . הגרדיינט של פונקציית המחר לפי הפרמטרים של ה- z -generator מוחושב באופן הבא:

$$\nabla_{\theta} V(G_{\theta}, D_{\phi}) = \frac{1}{m} \nabla_{\theta} \sum_{i=1}^m \log \left(1 - D_{\phi}(G_{\theta}(z_i)) \right)$$

וכאשר מאמנים את ה- z -discriminator נראה כך:

$$\nabla_{\phi} V(G_{\theta}, D_{\phi}) = \frac{1}{m} \nabla_{\phi} \sum_{i=1}^m \log D_{\phi}(x_i) + \log \left(1 - D_{\phi}(G_{\theta}(z_i)) \right)$$

נагו לבצע מודיפיקציה קלה על פונקציית המטרה של ה- z -generator. כיוון שבהתחלת הדגימות המיוצרות על ידי ה- z -generator לא דומות לחילוץ לאלו מסט האימון, ה- z -generator מזהה אותן בקטגוריות. כתוצאה לכך הביטוי $D(G(z))$ מקבל ערכים מאד קרובים ל-0, ומילא גם הביטוי $\log(1 - D(G(z)))$ מילא גם המטרה $D(G(z))$ קרוב ל-0. עניין זה גורם לכך שהגרדיינט של ה- z -generator גם יהיה מאד קטן, ולכן כמעט מטבע ולא מtbody שיפור ב- z -generator. לכן במקרה לחפש מינימום של הביטוי $\log(1 - D(G(z)))$ מוחפשים מינימום לביטוי $\log(D(G(z)))$. הביטויים לא שווים לגורם אר שיהם מוביילים לאוטו פתרון של בעיית האופטימיזציה אותה הם מייצגים, והביטוי החדש עוזב יותר טוב נורמי ומציל לשפר את ה- z -generator בזורה 'עליה יותר'.

הערכים האופטימליים של G ו- D :

כזכור, פונקציית המחר הינה:

$$V(D, G) = \min_G \max_D \mathbb{E}_{x \sim Data} \log D(x) + \mathbb{E}_{z \sim Noise} \log \left(1 - D(G(z)) \right)$$

כעת נרצה לחשב מה הערך האופטימי של ה- z -generator discriminator עבור generator נתון, ובוורו לחשב את הערך של פונקציית המחר. לשם הנוחות נסמן את התפלגות הדטה האמיתית ב- \mathcal{D} , ואת התפלגות הדטה הסינטטי המיוצר על ידי ה- z -generator ב- \mathcal{G} . עבור G קבוע, ניתן לרשום את פונקציית המחר כך:

$$V(D, G) = \int_x p_r(x) \log D(x) + p_g(x) \log(1 - D(x)) dx$$

כדי להביא את הביטוי זהה למקסימום, נרצה למקסם את האינטגרד עבור כל ערך x האפשרי. לכן הפונקציה לה מעוניינים למצואו אופטימום הינה:

$$f(D(x)) = p_r(x) \log D(x) + p_g(x) \log(1 - D(x))$$

נגזר את הביטוי האחרון ונשווה ל-0 כדי למצוא את הערך האופטימלי של $D(x)$ עבור x נתון:

$$\frac{\partial f(D(x))}{\partial D(x)} = \frac{p_r(x)}{D(x)} - \frac{p_g(x)}{1 - D(x)} = 0$$

$$\rightarrow p_r(x)(1 - D(x)) - p_g(x)D(x) = 0$$

$$D(x)_{opt} = \frac{p_r(x)}{p_r(x) + p_g(x)}$$

הביטוי שהתקבל הינו הערך האופטימלי של ה-discriminator generator קבוע (ביחס לקלט x נתון).ysis מושפע מהערך המקרה בו ה-GAN מצליח לייצר דוגמאות שנהרות אמיתיות לחלוין, כלומר $p_r(x) = p_g(x)$, אז מתקיים $D(x) = \frac{1}{2}$. הסתברות זו מושמעותה שה-discriminator לא יודע להיחלט לגבי הקטלט המתתקבל, והוא קבוע שהסתברות שהקלט אמיתי זהה לזה שהקלט סינטטי.

כעת נבחן מהו ערך פונקציית המחיר כאשר D אופטימלי:

$$\begin{aligned} V(G, D) &= \mathbb{E}_{x \sim Data} \log D(x) + \mathbb{E}_{z \sim Noise} \log(1 - D(G(z))) \\ &= \mathbb{E}_{x \sim Data} \log \left(\frac{p_r(x)}{p_r(x) + p_g(x)} \right) + \mathbb{E}_{z \sim Noise} \log \left(1 - \left(\frac{p_r(x)}{p_r(x) + p_g(x)} \right) \right) \\ &= \mathbb{E}_{x \sim Data} \log \left(\frac{p_r(x)}{p_r(x) + p_g(x)} \right) + \mathbb{E}_{z \sim Noise} \log \left(\frac{p_g(x)}{p_r(x) + p_g(x)} \right) \\ &= \mathbb{E}_{x \sim Data} \log \left(\frac{p_r(x)}{\frac{(p_r(x) + p_g(x))}{2}} \right) + \mathbb{E}_{z \sim Noise} \log \left(\frac{p_g(x)}{\frac{(p_r(x) + p_g(x))}{2}} \right) - \log 4 \end{aligned}$$

הביטוי המתתקבל הינו המרחק בין התפלגיות p_r ו- p_g , והוא נקרא Jensen-Shannon divergence ומוסמן ב- \mathcal{D}_{JS} . מרחוק זה הינו גרסה סימטרית של מרחק \mathcal{D}_{KL} (Kullback-Leibler divergence) בין התפלגיות P ו- Q , והוא מוגדר באופן הבא:

$$\mathcal{D}_{JS} = \frac{1}{2} \mathcal{D}_{KL}(P||M) + \frac{1}{2} \mathcal{D}_{KL}(Q||M), M = \frac{1}{2}(P + Q)$$

קייםנו שעבור D אופטימלי, פונקציית המחיר שווה למרחק \mathcal{D}_{JS} בין p_g עד כדי קבוע, ובאופן מפורש:

$$V(G, D_{opt}) = \mathcal{D}_{JS}(p_r, p_g) - \log 4$$

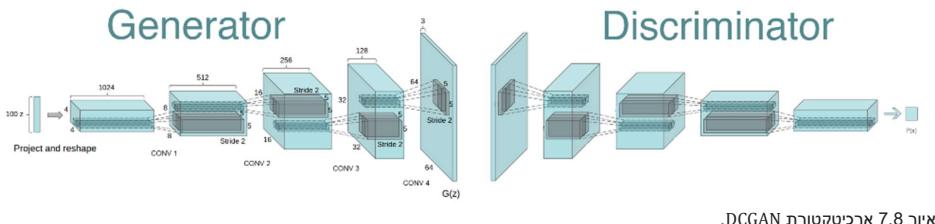
כאשר G אופטימלי ומתקיים $p_r(x) = p_g(x)$, אז המרחק בין התפלגיות שווה 0, כלומר $\mathcal{D}_{JS}(p_r, p_g) = 0$, ולכן:

$$V(G_{opt}, D_{opt}) = -\log 4$$

יש משמעות גדולה לביטוי שהתקבל – ככל שנצליח למשער יותר את $\mathcal{D}_{JS}(p_r, p_g)$, כך נצליח לקבל GAN יותר טוב.

7.2.2 Deep Convolutional GAN (DCGAN)

כפי שהוסבר בפרק 5, רשותות קוגנולוציה עלות יותר בדומין'י של תמונות מאשר רשותות FC. לכן הינה טبعי לחתות רשותות קוגנולוציה ולבנות בעדרתן generator ו-discriminator של תמונות. ה-generator מקבל וקטור אקראי ומוביל אותו דרך רשותת קוגנולוציה על מנת ליצור תמונה, וה-discriminator מקבל תמונה ומעביר אותו דרך רשותת קוגנולוציה שעשוה סיווג לבנייתו אם התמונה אמיתי או סינטטי. DCGAN נמצא ב-2015 ומאז פותחו רשותות שמייצרות תמונות יותר איקוטיות הן מבחינת הרזרולוציה והן מבחינת הדמיון שלהם להלן לתמונות אמיתיות, אך החשיבות של המאמר נעהוצה בשימוש ברשותות קוגנולוציה עבור GAN שימושי לדומין'י של תמונות.



איור 7.8 ארכיטקטורת DCGAN.

7.2.3 Conditional GAN (cGAN)

לעתים מודל גנרטיבי נדרש ליצור דוגמא בעלת מאפיין ספציפי ולא רק דוגמא שנראית אוטנטית. למשל, עבור אוסף תמונות המייצגת את הספסות 0-9, ונרצה שהGAN ייצור תמונה של ספרה מסוימת. במרקם אליל, בנוסח לוגיקטור הנכיהה, ה-GAN מקבל תנאן מסוים על הפלט אותו הוא צריך ליצור, כמו למשל ספרה ספציפית אותה רצים לקבב. GAN כזה נקראGAN (cGAN) או בקיצור conditional GAN (cGAN), ופונקציית המחיר שלו דומה מאוד לפונקציית המחיר של GAN רגיל למעט העבודה שהביטויים הופכים להיות מותנים:

$$\mathcal{L}_c(D, G) = \min_G \max_D \mathbb{E}_{x \sim Data} \log D(x|y) + \mathbb{E}_{z \sim Noise} \log (1 - D(G(z|y)))$$

7.2.4 Pix2Pix

כפי שראינו, ה-GAN הקלאסי שתוואר לעיל מסוגל ליצור דוגמאות חדשות מוקטור אקראי z, המוראל מהתפלגות מסוימת (בדרכ כל התפלגות אסוציאטיבית סטנדרטית), אך זה לא מוכחה. ישן גישות נוספת ליצור דטה חדש, כמו למשל ייצור תמונה חדשה על בסיס קוווי מתאר כללים שלה. סט האימון במרקם זה הביאה מזוגות של תמונות והסקציות שלן.

Pix2Pix משתמשת בארכיטקטורה של GAN אך במרקם חדש וktor z מהתפלגות כלשהי, הארכיטקטורה של מקלט סקיצה של תמונה בתווך קלט, וה-generator לומד להפוך את הסקיצה לתמונה אמיתי. הארכיטקטורה של discriminator נשארת ללא שינוי ייחוס מה שתוואר קודם לכן (פרט להסתrema לבינה הקלט), אך ה-generator מתקבל למשך זוג תמונות – את הסקיצה ואת התמונה (פעם כנ' משתנה – במרקם לקבל תמונה וביצוע סיווג בנאר), והוא מקבל זוג תמונות – את הסקיצה ואת התמונה (פעם כנ' משתנה – במרקם האימון המתאימה לסקיצה S ועם זאת שמייצרת על ידי ה-generator על בסיס S). על ה-tensor המתקבל האמם התמונה היא אכן תמונה אמיתי של הסקיצה או תמונה סינטטית. וריאציה זו של discriminator לקבוע האמם התמונה – כתעת ה-generator ציריך ללמידה שני דברים – גם ליצור תמונות טובות כך ששונה גם את פונקציית המחיר – כתעת ה-generator ציריך ללמידה שני דברים – וגם לטעות discriminator-ים יאמין שהן אמיתיות, וגם לפחות את המרחק בין התמונה שנוצרת לבין התמונה אמיתיות השicket לשקיצה.

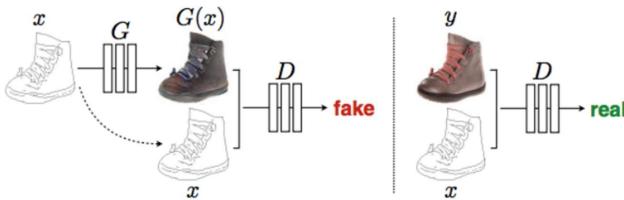
cuttn cross entropy ומרחק אוקליידי L_1 בין תמונת המקור לבין הפלט:

$$V(D, G) = \min_G \max_D \mathbb{E}_{x,y} (\log D(x,y) + \log (1 - D(x, G(x))))$$

$$\mathcal{L}_{L1}(G) = \min_{\theta_g} \mathbb{E}_{x,y} \|G(x) - y\|_1$$

$$\mathcal{L}(G, D) = V(D, G) + \lambda \mathcal{L}_{L1}(G)$$

ניתן להסתכל על pix2pix המפה תמונה לתמונה (image-to-image translation). נציין שבמקרה זה הקלט והפלט של pix2pix שייכים למתחומים (domains) שונים (סקיצה ומונה רגילה).

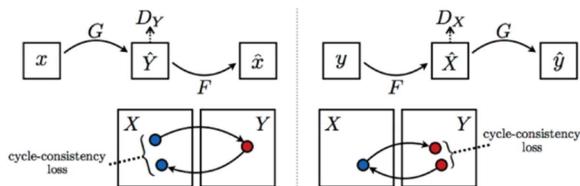


איור 7.9 ארכיטקטורת Pix2Pix

7.2.5 CycleGAN

ב-Pix2Pix הدادה המקורי הופיע בזוגות – סקיצה ואיתה תמונה אמיתי. זוגות של תמונות זה לא דבר כל כך ציוני, ולכן שיפרו את תהליכי האימון כך שיהיא ניתן לבצע אותו על שני סטים של דatasets מתחומים שונים. הארכיטקטורה עבורי המשימה זו מרכיבת משני generators – generators G ו- D – בהדרלה מכנים דוגמא מודולרי הראשוני x ל- G , שמנסה להפוך אותו לדוגמא מודולרי שני y , והפלט בכונס ל- D discriminator F שמנסה לשחרר את המקור x . המוצא של G -ה- y נכנס לא רק ל- F אלא גם ל- D_y discriminator D_y שנועד להנחות האם התמונה שהתקבלה הינה אמיתי או לא (עבור המודולן של y). ניתן לבצע את תהליכי היזה באופן דו-אי עבור y – מכנים את y ל- F על מנת לקבל את x ואת השמי F נועד לשפר את תהליכי הלמידה – לאחר ש- x הופך ל- y דרך G , ניתן לקבל חזרה את x אם עברו את y דרך F ($F(G(x)) \approx x$) (ה- F הותהיר של השוואת הכנסה למצאה נקרא *cycle-consistency*, והוא מושך עוד איבר לפונקציית המהיר, שמטרתו למחער עד כמה שניתן את המרחק בין התמונה המקורי לתמונה המשוחזרת:

$$V(D_x, D_y, G, F) = \mathcal{L}_{GAN}(G, D_y, x, y) + \mathcal{L}_{GAN}(F, D_x, x, y) \\ + \lambda (\mathbb{E}_x \|F(G(x)) - x\|_1 + \mathbb{E}_y \|G(F(y)) - y\|_1)$$

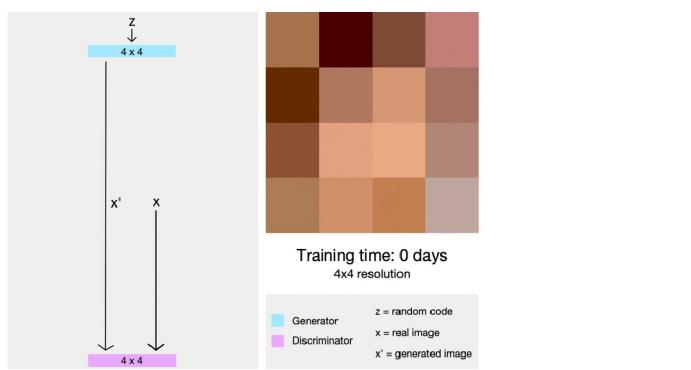


איור 7.10 ארכיטקטורת CycleGAN

7.2.6 Progressively Growing GAN (ProGAN)

כאמור לעיל, עבור דומין של תמונות, היגיון להשתמש ברשתות קובולוציה עboro יצירת תמונות חדשות, וזה הרעיון הבסיסי שמאחורי DCGAN. למוריה היכולה המרשימה של DCGAN ביצירה של תמונה באיכות גבוהה, יכולת זאת מוגבלת לתמונות בגודל מסוים. ככל שהרזולוציה של התמונה גבוהה יותר, כך יותר קל להבחין אם תמונה זו אמיתי או נזירה על ידי טכניקת גראטביה. בעוד ש-DCGAN מצליח ליצור תמונות שנאות וותניות בגדלים של 128 × 128, ואפיילו 32 × 32, 64 × 64, 32 × 32, 16 × 16, והוא מנסה ביצירת תמונות ברזולוציות גבוהות יותר, כמו למשל רזולוציה של 256 × 256 ProGAN. ProGAN הוא ה-GAN הראשון שפץ את מחסום הרזולוציה והצליח ליצור תמונות איקוטיות מאוד (במאמר המוקורי של ProGAN הוא שצלילהו ליצור תמונה בעלת רזולוציה גבוהה יותר מאשר התמונה原ала סינטטיות. אולם עוד לפני ProGAN הוצג GANs ProGAN המוקורי יצר רק למדוד שינויים תכונות של תמונות אחרות ברזולוציה גבוהה (2x2x2x2), אך זו משימה אחרת, מכיוון שבשבילו צריך רק למדוד שינויים תכונות של קלות, ולא ליצור תמונה חדשה לגמור מאפס).

הרעיון העיקרי מאחורי ProGAN, שהוצע ב-2017 על ידי חוקרים מחברת Nvidia, הינו לייצר תמונות ברזולוציה הולמת וגדלה בזרה הדרגתית. כמובן, מקום לסתות לאמן את השכבות של ה-generator ה- z -based במת אחת, כפי שנעשה בכל ה-GANs לפני כן, ניתן לאמן אותו לייצר תמונות ברזולוציה משתנה – בהתחלה הוא מותאם לייצר תמונות ברזולוציות מאד נמוכה (4 × 4), ולאחר מכן המשיכו לייצר תמונות ברזולוציה 8 × 8, אחר כך 16 × 16, 32 × 32, 64 × 64, 128 × 128 ועוד. וכך הלאה עד יצירה של תמונה ברזולוציה של 1024 × 1024.



איור 7.11 ארכיטקטורת ProGAN.

כדי לאמן GAN ליציר תמונות בגודל 4×4 , התמונות מסט האימון הוקטו לגודל זה (down-sampling). אחריו שה-GAN לומד לייצר תמונות בגודל 4×4 , מוסיפים לו עד שכבה המاضירה להכפיל את גודל התמונות המקוריות, קרי לייצור תמונות בגודל 8×8 . יש לציין שהאמון של הרשת עם השכבה הנוספת מהפיל עם המשקלים שאים נזדקם לכך, אך לא "מקפיאים" אותו, כזכור הם מעודכנים גם כן תוך כדי אימון הרשת בשבייל לייצור תמונה בחולצה כפולה. הгалלה הדרגתית של הרוחולציה מאלצת את הרשת לומדת "לבצע" לביצוע "הagtoms" של התמונה (פפואים בתמונה מוטשטשת מאוד). לאחר מכן מכך הרשת "לומדת" ליציר את הרוחולציה (הagtoms) של התמונה המוטשטשתות האלה. תהילך זה משפר את איכות התמונה הסופית כיון שבאופן זה סבירות שהרשת תלמד דפוסים משמעותיים קטנה ממשמעותית.

7.2.7 StyleGAN

StyleGAN, שיצא בשליה בשנת 2018, מציע גרסה מושדרגת של generator, עם דגש על רשת ה-*latent*. המאמר שמו לב כי היתרונו הפטנטניאלי של שכבות ProGAN המייצירות תמונה בצורה הדרגתית נבע מיכולתו לשילוט בתוכנות (מאפיינים) ויזואליות שונות של התמונה, אם משתמשים בהן כראוי. ככל שהשכבה והרוחולציה מוגהה יותר, כך התוכנות שהיא משפיעה עליה גסות יותר.

למעשה, הינה ה-*GAN* הראשון שנותן יכולת לשילוט במאפיינים ויזואליים (אומנם לא בצורה מלאה) של התמונה הנוצרת. מחברי StyleGAN חילקו את התוכנות הייזואליות של תמונה ל-3 סוגים:

- גס: משפיע על תנוחה, סגנון שיר כליל, צורת פנים וכו'.
- אמצעיות: משפיע על גווני פנים יותר, סגנון שיר, עיניים פקוחות/עכומות וכו'.
- רוחולציה דקה: משפיע על צבע (עיניים/שיר/עור) ועל שאר תכונות המיקרו של תמונה.

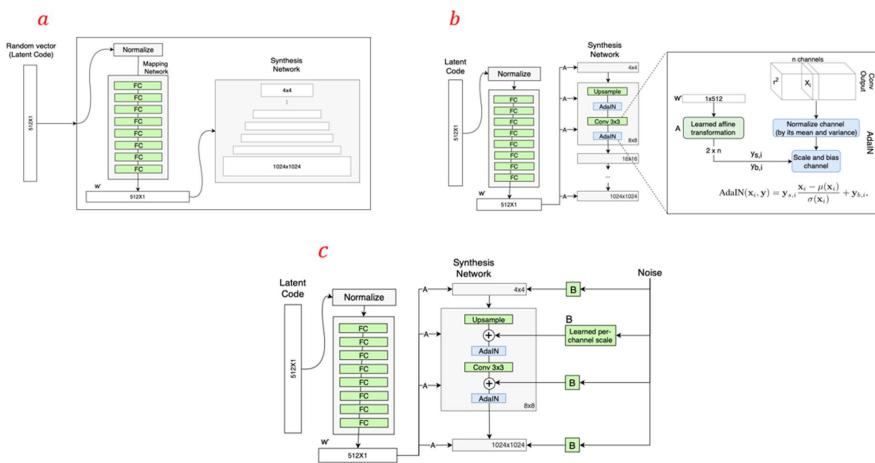
כדי להעניק ל-*StyleGAN* את היכולות הללו, נדרש מספר שינויים ביחס לארכיטקטורה של ProGAN (נתאר רק את שלושת השינויים החשובים ביותר כאן):

הוספה רשת מייפוי: מטרת רשת המיפוי היא קידוד וקטור הקטלט לווקטור ביניים \mathbf{w} (הנקרא וקטור סגןון) אשר האיברים השונים שלו שלטים בתוכנות ויזואליות שונות של התמונה הנוצרת. זה הוליך לא טריוויאלי McCoyו שהיכולת של הרשת לשילוט בתוכנות ויזואליות באמצעות וקטור הקטלט נזק. הסיבה לכך טמונה בעובדה שווקטור הקטלט נאלץ "עקבות אחר ציפוי" הסת坐着 של סט האימון" שגורם לתופעה הנקראית (FE) – ערבוב מאפיינים. FE בין תכונות צבע השיר והמגדר יכול להופיע גם למשול סט או אימון שיש מגמת כלית של גברים עם שער קצר ונשים בעלות שער ארוך. במקרה זה הרשת תלמד שגברים יכולים להיות בעלי שיר קצר בלבד ולהיפך אצל נשים. כתוצאה לכך, אם "נשחק" עם רכבי וקטור הקטלט כדי לייצר תמונה של גבר בעל שיר ארוך, בסופו של דבר מגדרו ישנה גם כן ותקבל תמונה של אשה.

רשת המיפוי שהווסף לארכיטקטורה הופכת את וקטור הקטלט לווקטור ביניים \mathbf{w} שאינו צריך לעקוב אחר התפלגות של סט האימון, ורק יש פחות ערבוב המאפיינים. במקרים אחדות, רשת זו מאפשרת את היכולת לשילוט במאפיינים ויזואליים של התמונה הנוצרת באמצעות שני רכיביו של וקטור \mathbf{w} . רשת המיפוי מורכבת ממשונת שכבות FC וגודל הפלט שללה זהה לגודל הקטלט.

החלפת BN ב-AdaIN: רשותת הקונבולוציה של ה-*generator*, שנעודה לייצר תמונות ברזולוציות שונות משלستخدم במנגן שנקרא AdaIN (Batch Normalization). בשונה מ-BN, הפרמטרים של הממוצע ושל השונות בגשת N ננדלים מוקטור הסוגנון \mathbf{w} (המוצג טרנספורמציה לינארית של \mathbf{w} עם משקלים נלדיים). להבדיל מ-AdaIN, במנגן BN סטנדרטי פרמטרים אלו ננדלים כמו המשקלים الآחרים ולא תלוים במצבם של שכבה כלשהי.

ויתור על אקריאיות של וקטור קלט: ב-StyleGAN וקטור הקלט אינו וקטור המוגדר מהתפלגות גאוסית אלא וקטור דטרמיניסטי עם רכיבים נלדיים. וקטורי הרעש מתווסףים ישירות לפטלים של ערוץ קונבולוציה ברשומות ה-*generator* כאשר העוצמה שלהם ננדמת לכל ערץ בנפרד. שימוש בוקטור קלט דטרמיניסטי במנגן בוקטור אקראי מקל לכלי הנגראה על הפרדת המאפיינים על ידי רשת המיפוי (ויתר קל לעשרות זאת על וקטור קבוע מאשר את משקליו רשת המיפוי לווקטור כוכב אקראיים).



איור 7.12 השלדים העיקריים בארכיטקטורת מייפוי. (a) הוסיף רשת מייפוי. (b) שימוש ב-AdaIN. (c) שימוש ב-StyleGAN. יש עוד כמה שינויים יותר מינוחרים ב-StyleGAN, כמו שינוי של היפר פרמטרים של הרשותות, פונקציית מחיר וכו'. התוצאות הן לא פחות ממרשים – StyleGAN יוצר תמונות שנראות ממש אמתיות ובנוסף מקנה יכולת לשילוט בחלק מהתכונות החזויות של התמונות.



איור 7.13 תמונות שיוצרים באמצעות StyleGAN.

7.2.8 Wasserstein GAN

הנחה היסודית ברוב המודלים האנרגטיביים, ובפרט ב-GANs, הינה שהדתא הרב ממד' (למשל תמונה) "ח' במשתח מממד נמוך בתוכו. אפשר להסתכל על משפטה בתור הכללה של ת-מרחב וקטורי ממוקן הנפרש על ידי ת-מרחב קבוצה של וקטורי בסיס של מרחב וקטורי מממד גבוה יותר. גם המשפט נוצר מת-מרחב קבוצה של וקטורי הבסיס של "מרחב האם", אך ההבדל בין ת-מרחב וקטורי מתבצע בכך שימושה עשויה להיות צורה מאוד מורכבת יחסית לת-מרחב וקטורי. משטענו מכך שניתן ל"יציר" דאטה רבת ממד' על ידי טרנספורמציה של וקטורי מרחב בעל ממד נמוך (וקטור לטנטו). למשל, ניתן בעדרת רשף ווירטואלי ל"יציר" תמונה בגודל $3 \times 64 \times 64$ פיקסלים מוקטור באורך 100 בלבד. זאת אומרת,سام התפלגות התמונה של הרשות הגיגנטית וגם התפלגות של הדטה האומני נצעדים ב- "משתח בעל ממד נמוך" בתור מרובה בעל ממד גבוה של הדטה המקורי. באופן פרטני יותר, משטח זה נקרא ריעעה (manifold), וההשערה שתויה מעלה מהוה הנחת יסוד בתחום גראן-קלילדי רישעות (manifold learning). מכיוון שמדובר במושגים בעלי ממד נמוך בתור מרחב בעל ממד גבוה, קיימות סכירות גבואה שלא יהיה שום חיטין בין המשטח בו "ח' הדטה האומני לבין זה של הדטה המקורי" (כל הפחות בתחום תחילה האימון של (GAN)). יתרה מכך, המרחק בין משטחים אלה עשוי להיות די גדול. מכך נובע שה- גזול בין D discriminator לשערת אמינותית ל- "סינטטי" בקבוקת. בעודו, קיון שמרחב מממד גבוה יש מרחק ממשקרים לאפס כי אכן קל להיות רוחקים מאד אחד מהשני.

רקע זה מוסייע להבין מדוע הפער שיש בין ה-generator וה-discriminator מבחינת אופי הלמידה בעיה. כאמור, generator, ה-discriminator שלו על סמך הצלויים שהוא מקבל מה-generator (discriminator) (discriminator) (GAN). אבל אם generator, discriminator פשטוט לא יכול לשפר את איכות התמונהות שהואר מעלה) לדוגמאות המיצירות על ידי generator, discriminator, ה-shallow generator טוב יותר יחסית ל-G". אתגר זה בא לידי ביטוי גם במצבה של שהוא מייצר. במקרים מסוימים, D" שפスト הרבה יותר טוב יחסית ל-G". generator discriminator-ל-discriminator.

יש מספר לא-קסון של שיטות הבאות לשפר את תהליכי האימון של GAN, אך אף אחת מהן אינה מטפלת בבעיה זו באמצעות شيء של פונקציית המחיר. השיטות הבולטות הן:

- התאמת פיצ'רים (feature matching))
 - minibatch discrimination
 - virtual batch normalization
 - מינימום היסטורי (histogram minimum)

כפי שהואסביר, הבעה של המרחק בין היריעות משתקפת במבנה של פונקציית המחיר, $\psi_{\text{סכום}}$, ניתן לנוסח ולפתור את הבעה מהוורש על ידי שימוש בפונקציית מחיר יותר מתאימה. לשם כך ראשית ונמן את התפלגות הדadata האמצעי-ב- k , ואת התפלגות הדאטאות הסינטטיים מיצורי על ידי $\psi_{\text{generator-}b-g}$. לעלי הריאינו שפונקציית המחיר האופטימלית המיצעת את המרחק בין התפלגיות $\psi_{\text{ס.}}$ מתחזרת על ידי D_{JS} – Jensen-Shannon divergence.

נית להוכיח כי מרחק D_J בין ההתפלגויות p_r, p_g לא ניתן לשוניים ב- \mathcal{C} כאשר המשתוחים שבמה "ח'ם" $p_g - p_r$ יוכיחים אחד מהשני. לעומת זאת, מרחוק D_J כמעט ולא ישנה אחרינו עדכון משקלים של ה- generator , וממילא לא ישקר את המרחק המעדכן בין שתי התפלגויות p_r, p_g . זו למעשה הביעה המוחשית ביותר כי יותר עם פונקציית המהירות המוקנית של ה- generator , שעדכון המשקלים לא משיפוי כמעט על D_J , יכול שומרה התפלגויות p_r, p_g רחוקות אמתם האובייקטיבית.

Wasserstein GAN בא להתמודד עם בעיה זו, ולשם כך הוא משתמש בפונקציית מחיר אחרת, בה עדכון המשקלים של generator גם במרקח בין ה- p_r -ות p_g . פונקציית המחר החדשה מבוססת על מරחק והקרא (EM) המהווה מקרה פרטי של מרחב וורשטיין המסומן ב- \mathcal{W} . מרחב וורשטיין מסדר $1 \geq k$ בין שתי מידות הסתברות x, μ על מרחב M מוגדר באופן הבא:

$$W_p(\mu, \nu) = \inf_{\gamma \in \Gamma(\mu, \nu)} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|] = \left(\inf_{\gamma \in \Gamma(\mu, \nu)} \int_{M \times M} d(x, y)^p d\gamma(x, y) \right)^{\frac{1}{p}}$$

כאשר (μ, ν) הן כל מידות הסתברות על מרחב המכפלה (product space) של M עם עצמו (זהו למשה מרחב המכיל את כל הזוגות האפשריים של האלמנטים M -ים M) עם פונקציות שוליות (marginal) μ, ν בהתאם. תחת סימן האינטגרל יש את המרחק האוקלידי מסדר k בין הנקודות. מרחב EM הינו מקרה פרטי של מרחב וורשטיין, כאשר $k = 1$, ובאופן מפורש:

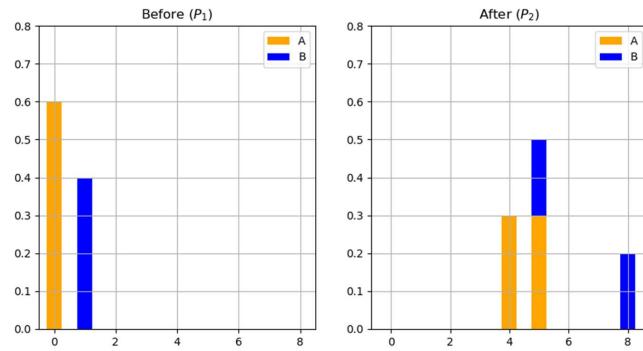
$$EM = W_1(\mu, \nu) = \inf_{\gamma \in \Gamma(\mu, \nu)} \int_{M \times M} d(x, y) d\gamma(x, y)$$

הגדרה זו נראית מאוד מסובכת וננסה לתת עבורה אינטואיציה, ולהבין מדוע עבור $k = 1$, מרחב וורשטיין נקרא EM. לשם הפשטות נניח שהמרחב M הינו חד ממדי, כלומר קי ישר, ועליו עשר משקלות של 0.1_{kg} כל אחת 6 משקלות $x = 0$ בנקודת (0.6_{kg}) , 4 משקלות $x = 1$ בנקודת (0.4_{kg}) , 5 משקלות $x = 2$ בנקודת (0.3_{kg}) , 8 משקלות $x = 3$ בנקודת (0.2_{kg}) , ועוד 0.5_{kg} שאר המשקלות $x = 4$ בנקודת (0.1_{kg}) .

כמובן הרבה דרכיים לבצע את היזזת המשקלות, ונרצה למצוא את הדרך העילית ביותר. לשם כך נגידirm מאץ מכפילה של משקל במרקח אותו מזידם את המשקל (בפייזיקה ממשga זה נקרא עבודה - כוח המפעעל על הגוף לאורכו מסלול). בדוגמה המובאת, המאמץ המינימלי מתאפשר על ידי היזזת המשקלות באופן הבא:

$$\begin{aligned} & 0.3_{kg} \text{ מועברים מ-} x=0 \text{ ל-} x=4, \text{ כאשר המאמץ הנדרש לכך הינו } 1.2 = 0.3 \cdot (4-0). \\ & 0.3_{kg} \text{ מועברים מ-} x=0 \text{ ל-} x=5, \text{ כאשר המאמץ הנדרש לכך הינו } 1.5 = 0.3 \cdot (5-0). \\ & 0.2_{kg} \text{ מועברים מ-} x=1 \text{ ל-} x=5, \text{ כאשר המאמץ הנדרש לכך הינו } 0.8 = 0.2 \cdot (5-1). \\ & 0.2_{kg} \text{ מועברים מ-} x=1 \text{ ל-} x=8, \text{ כאשר המאמץ הנדרש לכך הינו } 1.4 = 0.2 \cdot (8-1). \end{aligned}$$

$$\text{סך המאמץ המינימלי שווה במרקחה זה: } 1.2 + 1.5 + 0.8 + 1.4 = 4.9.$$



איור 7.14 היזזת משקלות באופן אופטימלי. P_1 מייצג את המצב ההתחלתי, ו- P_2 הינו המצב לאחר היזזת המשקלות.

icut, במקום להסתכל על משקלים, נתיחס להתפלגויות p_1, p_2 , המוגדרות באופן הבא:

$$p_1(x) = \begin{cases} 0.6, & x = 0 \\ 0.4, & x = 1 \\ 0, & \text{else} \end{cases}, \quad p_2(x) = \begin{cases} 0.3, & x = 4 \\ 0.5, & x = 5 \\ 0.2, & x = 8 \\ 0, & \text{else} \end{cases}$$

השאלה כיצד ניתן להעיבר מסה הסתובבותית מ- m -ק' לשתකבל ההתפלגות k_2 , שקופה לדוגמא של ההזאות המשקולות. מרחוק EM בין שני התפלגויות k_1 , k_2 , מוגדר להיות ה-"מאזן" המינימלי המדרש בשבייל להעיבר את המסמה הסתובבותית מ- m_1 ל- m_2 , או במילים אחרות – מרחוק EM מגדיר מהי כמות ה-"יעבודה" (מאזן) המינימלית הנדרשת בשבייל להעיבר מ- m_1 ל- m_2 . אם נזהור לדוגמא של משקלות, נוכל להסביר מודע D_{ij} עבור $i = k$ ו- $j = k'$ מרחוק EM – Earth Mover's Distance – מרחוק בין שתי התפלגויות שקול לכמה אמצעי דריש להעיבר כמות אדמה ובמשקל מסוים כדי לעבר מחלוקה מסוימת של אדמה להקלחה אחרת. באופן יותר פורמלי – מידת ההסתובבות על מרחב המכפלות בנוסחה של מרחוק EM מוגדרת שאנו שbow אונתו מעברים את הסכמה הסתובבותית (משקל מסוים של אדמה), כאשר הביטוי (x,y) מציין כמה מסה הסתובבותית מועברת מנוקודה x לנוקודה y .

על מරחיק EM, ניתן לדעת עד כמה עדכון המשקלים מקרב או מרחקי את p_g מ- p_r .
 משתמשים פונקציית המחרה הקווית הנגדית באמצעות \mathcal{D}_{JS} . כתוב, בעדות פונקציית המחרה החדשה המבוססת
 של הדטה הסינטטי, נוכל לדעת בערךת מරחיק EM עד כמה השנה המחרה קיימת בינו לבין היריעות. נציין שזה לא קורא כאשר
 EM דע לשער בזרה טבה את הרווח שבין "חו"ת" שמי התפלגותים, כלומר אם מעריכים את היריעה
 בסופר מארק**D_W** בין התפלגות המחרה במרחיקים לבין בקבוקיות אלו. תכונה זו היא למעשה בדיקת מה שצ'יר
 פונקציית מחרק **D_W** בין מידות ההסתברות המתחשב בתוכנות של הקבוקיות עליהן מידות אלו
 לאחר שהוסבר מהו מוחק וסורתטי **D_W** ומהו מרחיק EM. ניתן להבין כיצד ניתן להשתמש במושגים אלו עבור

באופן תיאורתי זה המוצע, אך עדין זה לא מספק, כיון שהוא מנסה למסור דרך לחישוב את D_w , או לכל הפחות את מרחק $EM = d$, ככלומר את מרחק EM . במקור מרחוק זה מוגדר כבעית אופיינית של מידות הסטברות על מרחב המבנה, ציריך למצאו דרך להשתמש בו כפונקציית מרוח. בששל לבצע את ניתן להשתמש בפורה על מרחב המבנה, (*Rubinstein-Kantorovich*) RK, לפיו ניתן לחישוב את D_w , $p = 1$ באופן הבא:

$$W(p_r, p_g) = \frac{1}{K} \sup_{\|f\|_L} E_{x \sim p}[f(x)] - E_{x \sim p_s}[f(x)]$$

ב-*(x)* הינה פונקציית **K-ליפשיץ רציפה** (כלומר, פונקציה רציפה עם קצב השתנות החסום על ידי K). ב-*(y)*Cut נניח ש-*(w)* הינה פונקציית **K-ליפשיץ רציפה** המוגדרת discriminator בעל סט הפרמטרים w . ה-*(z)* מוחשב באמצעות מנגנון דומה, אך הוא מוגדר כפונקציית discriminator בעל סט הפרמטרים z .

$$L(p(r), p(g)) = W(p(r), p(g)) = \max_{w \in \mathcal{W}} \mathbb{E}_{x \sim p_r} [f_w(x)] - \mathbb{E}_{z \sim p_r(z)} [f_w(g_\theta(z))]$$

פונקציית מהיר זו מודדת את המרחק \mathcal{D}_W בין ההתפלגות p_g, r_g , וכך שפונקציה זו מקבל ערכים יותר נומקיים מכמה generator-discriminator. ניתן ליצר דוגמאות שמתפלגות באופן יותר דומה לדאטה המקורי. בשונה מ-GAN קלאס'ר בו-

להבחן בין דוגמא אמיתית לסינטטי, אלא מזמין ללמידה פונקציית K-ליפשיץ רציפה המודדת את \mathcal{D}_W בין ההתפלגות p_r, p_g . generator-leuometria זאת אמונה למצער את $L(p_r, p_g)$ כאשר רק האיבר השני שתלו ב- p_g .

וככל שפונקציית המהיר הולכת וקטנה, כך מתקרב יותר L_{p_r} .

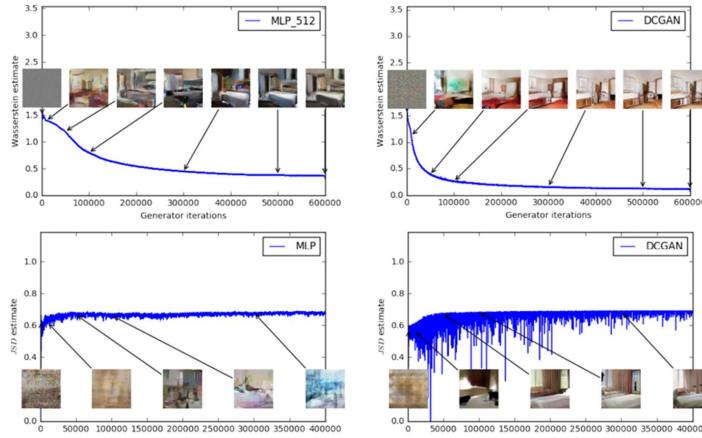
כאמור, תנאי הרכח לשימוש במרקח זה בפונקציית המחיר הינו שהפונקציה תהה K-יליפשיץ רציפה. מסתבר שקיים תנאי זה אין ממשימה קלה כלל. כדי להבטיח את קיומו, המאמר המזכיר הציג עלוצץ קיטימה של משקל'י ה- ℓ -discriminators לוטשו סופ' מוסים, נניח [0.01, 0.01]–[]. ניתן להראות כי קיטימה זו מבטיחה את ש- f_w תהיה K-יליפשיץ רציפה. אולם, כמו שראינו המאמר מודדים בעוצם, ביעוץ אחד קיטימה בכדי לDAOש ש- f_w יישר יכול גורם לבלעויות אחרות. למשל, כאשר החלן והקטינה הושם על המשקל'ים צור מז', הגדר'אנטס של Wasserstein GAN עלילים ללהתאפס, מה שיאט את התייר הלמידה. מצד שני, כאשר החלן זה הרבה יותר, ההתקנות עלולות להיות מאוד איטיות.

האימון של GAN Wasserstein דומה לאימון של ה-GAN המקורי, למעט שני הבדלים עיקריים:

- א. קיצוץ טווח המשקלים על מנת לשמור על רציפות-ליפשייז.
 ב. פונקציית מחירי המסתמכת על \mathcal{D}_W במקומם על \mathcal{D} .

תהליך הלמידה מתבצע באופן הבא – לאחר כל עדכון משקלים של ה-discriminator (gradient ascent) מתקיימים את טוויה המשקלים. לאחר מכן מבוצעים עדכון רגיל של משקליו ה-generator (gradient descent).

Wasserstein GAN מצליח לאגורם לכך שהקורסילציה בין אינטואיטיבית הנוצרת על ידי ה-generator נובע ערך של פונקציית לוס תהיה הרבה יותר בולטות מאשר ב-GAN רגיל בעל אותה ארכיטקטורה. ניתן להמחיש זאת היטב באמצעות גורמים הבוחנים את היחס בין D_W לבין D_S :



איור 7.15 שעורק מרחק \mathcal{W} בין \mathcal{G}_r ל- \mathcal{G}_g כפונקציה של מספר האיטרציות (בגרפים העלונים), לעומת שערן מרחק \mathcal{D}_{JS} בין \mathcal{G}_r ל- \mathcal{G}_g כפונקציה של מספר האיטרציות (בגרפים התחתוניים).

ניתן לראות בבירור כי ככל שאיקות התמונה ש-generator מייצר עולה, כך \mathcal{W} הולך וקטן, ואילו מרחק \mathcal{D}_{JS} לא מראה שום סימן של ירידה. הצלחה זו נובעת מהשינוי בפונקציית המבחן, שגרם לאימון להיות יותרiesel, והביא לכך שהדוגמאות הסינטטיות תהינה דומות הרבה יותר לאטעה המקורית.

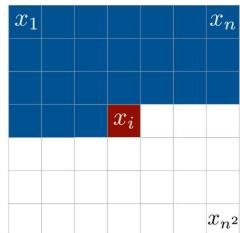
נקודה נוספת שיכולה להסביר את הצלחה החישובית של השימוש ב- \mathcal{W} נובעת מכך שמטריקת ורסטיין חלשהיחסית למטריקת \mathcal{JS} , וננסה להבהיר נקודה זו.

באופן אידיאלי, כאשר אנו מאמנים מודל היינו רוצחים להיות בטוחים שאם נגבה בצוואר ונגבה בפה וכל עוד נעדק המודול בדוחין על פי הוראות הגדר'יאט, נסימן את האימון בנקודה מסוימת אופטימלית. אולם בפועל זה לא תמיד כך, כיון שישנן בעיות שעוברן מטריקות מסוימות יגיעו למינומה זו ואחרות לא. ניקח לדוגמא שני אנשים שעומדים על סף תחום ורוצחים להגיע לעמק. האחד מזודד את הגובה ונתקדם על פיו, וכך הוא יגיע לסתה בקבלה יהיסטי. الآخر מתעטן במינומו על ציר פקoon דרום, וכך הוא עשוי ליתקל בקשטים במהלך הירידה, וגם אם הוא אכן יגיע למטרה, זה בהכרח יהיה בתהילן איטי יותר. באופן דומה, כאשר ללקחים זוג מטריקות, באופן פורמלי ניתן להגיד שאם התכונות של סדרת התפלגות תחת מטריקה אחת גוררת התכונות של הסדרה תחת מטריקה אחרת, אז המטריקה הראשונה חזקה יותר מהמטריקה השניה. העבודה ש- \mathcal{W} חלש יותר מר- \mathcal{D}_{JS} בכך אומerta שיטקן ויש בעיות שעבורן מתקבלות תוצאות אופטימלית עבור \mathcal{W} אך לא עבור \mathcal{D}_{JS} .

7.3 Auto-Regressive Generative Models

משוכה נוספת של מודלים גנרטיביים נקראת Auto-Regressive Generative Models, ובדומה לו-VAE, מודלים AR מוצאים התפלגות מפורשת של מרחיב מסוים ובעזרת התפלגות זו מיצרים DATA חדשה. עם זאת, בעוד VAE מוצאים קירוב להתפלגות של המרחב הלטני, שיטות AR מנסות לחשב במידוק התפלגות מסוימת, וממנה לדגום וליצור DATA חדש.

תמונה x בגודל $a \times a$ היא למעשה רצף של a^2 פיקסלים. כאשר רוצhim לייצר תמונה, ניתן ליצור כל פעם כל פיקסל בהתאם לכך שהוא תלוי בכל הפיקסלים שלפניו.



איור 7.15 תמונה כרצף של פיקסלים.
כל פיקסל הוא בעל התפלגות מותנית:

$$p(x_i|x_1 \dots x_{i-1})$$

כאשר כל פיקסל מורכב משולשה צבעים (RGB), וכן ההסתברות המדויקת היא:

$$p(x_{i,R}|x_{<i})p(x_{i,G}|x_{<i}, x_{i,R})p(x_{i,B}|x_{<i}, x_{i,R}, x_{i,G})$$

כל התמונה השלמה היא מכפלה הסתברויות המותניות:

$$p(x) = \prod_{i=1}^{n^2} p(x_i) = \prod_{i=1}^{n^2} p(x_i|x_1 \dots x_{i-1})$$

הביטוי (x) הוא ההסתברות של דיאט מסוים ליצג תמונה אמיתית, וכך נרצה למקסם את הביטוי הזה כדי לקבל מודל שמייצג תמונות שנראות אוטנטיות עד כמה שניין.

7.3.1 PixelRNN

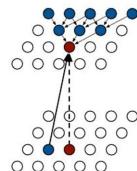
אפשרות אחת לחשב את (x) היא להשתמש ברכיבי זיכרון כמו LSTM עבור כל פיקסל. באופן טבעי יימנו רצויים:

$$\text{Hidden State } (i,j) = f(\text{Hidden State } (i-1,j), \text{Hidden State } (i,j-1))$$

הבעיה בחישוב זה היא הזמן שהוקח לביצועו. כיוון שכל פיקסל דרוש לדעת את הפיקסל שלפניו – לא ניתן לבצע אימון מקבילי לרכיבי LSTM. כדי להתגבר על בעיה זו חוץו כמה שיטות שנעמדו לאפשר חישוב מקבילי.

Row LSTM

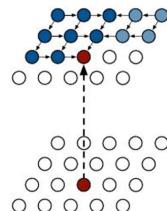
במקום להשתמש במצב החבוי של הפיקסל הקודם, ניתן להשתמש רק בשורה שמעל הפיקסל אותו רצויים לחשב. שורה זו עצמה מחושבת לפני כן על ידי השורה שמעליה, ובכך למעשיה לכל פיקסל receptive field של מושלש. בשיטה זו ניתן לחשב באופן מקבילי כל שורה בנפרד, אך יש לך מחריר של איבוד הקשר בין פיקסלים באותה שורה (loss context).



איור 7.16 – כל פיקסל מחושב על ידי $3 \times k$ פיקסלים בשורה שמעלוי.

Diagonal BiLSTM

כדי לאפשר גם חישוב מקבילי וגם שמירה על קשר עם כל הפיקסלים, ניתן להשתמש ברכיבי זיכרון דו כיווניים. בכל שלב מחשבים את רכיבי הזיכרוןensi הקיימים הנוכחיים, וכן כל פיקסל מחושב גם עזרת הפיקסל שלידיו וגם על ידי זה שמעלוי. באופן זה ה-*receptive field* גדול יותר וכן loss context, אך החישוב יותר איטי מהשיטה הקודמת, כיוון שהשורות לא מחושבות בפעם אחת אלא כל פעם עם שני פיקסלים.

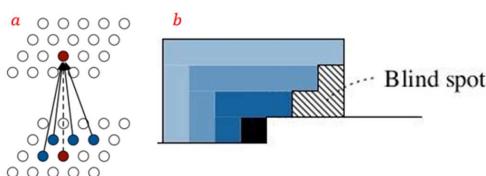


איור 7.17 – כל פיקסל מחושב על ידי ≥ 3 פיקסלים בשורה שמעלי.

כדי לשפר את השיטות שימושות ברכיבי זיכרון ניתן להוסיף עוד שכבות, כמו למשל Residual blocks שעוזרים להציג את ההתכוונות Masked convolutions- של הערכים השונים של כל פיקסל.

7.3.2 PixelCNN

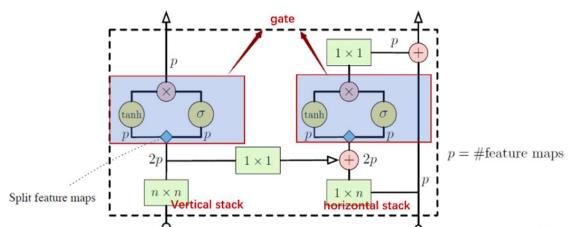
החישון העיקרי של PixelRNN נובע מהאמון האיטי שלו. במקום רכיבי זיכרון ניתן להשתמש ברשת קובולוציה, ובכך להציג את תהליכי הלמידה ולהגדיל את ה receptive field . גם בשיטה זו מתחילה מהפיקסל הפינתי, רק כעת הלמידה היא לא בערצת רכיבי זיכרון אלא באמצעות שכבות קובולוציה. היתרונות של שיטה זו על פניהם מرتبطים בקיורו משמעותי של תהליכי האימון, אך התוצאות פחות טובות. חיסרון נוסף בשיטה זו נובע מהמבנה של המנסנים | h-field – כל פיקסל מתחסס על שלושה פיקסלים שמעליים, והם בתורם כל אחד תלוי בשלושה פיקסלים בשורה שמעל. מבנה זה מנתק את התלות בין פיקסלים יחסית אך אינם ב- receptive field .



איור 7.18 (a) החישון של PixelCNN – ניתוק בין פיקסלים יחסית קרובים. (b) היחסון receptive field (b) – ניתוק בין פיקסלים יחסית קרובים.

7.3.3 Gated PixelCNN

בדי להתגבר על בעיות אלו – ביצועים לא מופיעים טובים והתעלמות מפיקסלים יחסית קרובים שאינם ב- receptive field – נעשה שימוש ברכיב זיכרון הדומה ל-LSTM, המשלב את רשותות הקובולוציה בתוך RNN .



איור 7.19 שכבה של Gated PixelCNN .

כל רכיב זיכרון בני משני חלקים – horizontal stack and vertical stack – כאשר כל אחד מהם הוא למעשה מעשה שכבת קובולוציה. ה- vertical stack בנוי מזיכרון של כל השורות שהו עד כה בתמונה, וה- horizontal stack הוא מסנן יחיד על הקטל הנוכחי. ה- horizontal stack עבר דרך שער של אקטיביזציה לא-ליניארית ובסיום מתחבר ל- vertical stack , כאשר גם החיבור ביניהם עבר דרך שער של אקטיביזציה לא-ליניארית. לפני כל כניסה של stack לתוך שער, המנסנים מופצלים – חצ' עוביים דרך tanh וחצי דרך סיגמאיד. בסך הכל המוצא של כל שער הינו:

$$y = \tanh(w_f * x) \Theta \sigma(w_g * x)$$

7.3.4 PixelCNN++

שיפור אחר של PixelCNN הוצע על ידי AI OpenAI , והוא מבסס על מספר מודיפיקציות:

- שכבה ה- SoftMax שקובעת את צבע הפיקסל זורכת הרבה-Calculations, כיוון שיש הרבה צבעים אפשריים. בנוסף, היא גורמת לארדיאנט להתאפשר מהר. כדי להתגבר על כך ניתן לבצע דיסקרטיזציה לצבעים, ולאפשר טווח צבעים קטן יותר. באופן זהה קל יותר לקובע את ערך של כל פיקסל, ובנוסף תהליכי האימון יתור יעיל.
- במקרים לביצוע בכל פיקסל את ההתניה על כל צבע בנפרד (כפי שהראינו בפתחה), ניתן לבצע זאת הפתינה על כל הצבעים יחד.
- אחד האתגרים של PixelCNN הוא יכולת המוגבלת למצוא תלויות בין פיקסלים רחוקים. כדי להתגבר על כך ניתן לבצע sampling down , ובכך להפחית את מספר הפיקסלים בכל מסנן, מה שמאפשר לשמר את הקשרים בין פיקסלים בשורות ורחוקות.

- בדומה ל-Net-U, ניתן לבצע חיבורים בעזרת Residual blocks ולשמור על יציבות במהלך הלמידה.
- שימוש ב-Dropout לצורף רגולרייזציה והימנעות מ-fitting.

7. References

VAE:

<https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73>

<https://jaan.io/what-is-variational-autoencoder-vae-tutorial/>

<https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html>

GANs:

<https://arxiv.org/abs/1406.2661>

<https://arxiv.org/pdf/1511.06434.pdf>

<https://phillipi.github.io/pix2pix/>

<https://junyanz.github.io/CycleGAN/>

<https://arxiv.org/abs/1710.10196>

<https://arxiv.org/abs/1812.04948>

<https://towardsdatascience.com/explained-a-style-based-generator-architecture-for-gans-generating-and-tuning-realistic-6cb2be0f431>

<https://arxiv.org/abs/1701.07875>

AR models:

<https://arxiv.org/abs/1601.06759>

<https://arxiv.org/abs/1606.05328>

<https://arxiv.org/pdf/1701.05517.pdf>

<https://towardsdatascience.com/auto-regressive-generative-models-pixelrnn-pixelcnn-32d192911173>

https://wiki.math.uwaterloo.ca/statwiki/index.php?title=STAT946F17/Conditional_Image_Generation_with_PixelCNN_Decoders#Gated_PixelCNN

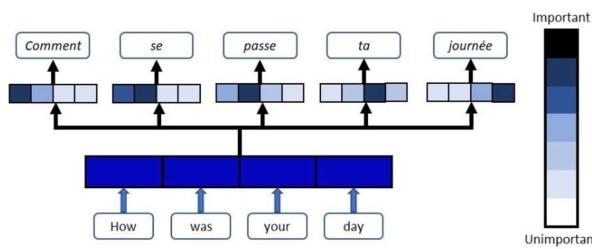
8. Attention Mechanism

8.1 Sequence to Sequence Learning and Attention

8.1.1 Attention in Seq2Seq Models

ניתוח סדרות בין ש קשר בין האיברים יכול להיעשות בעזרת רשות עם ריכבי זיכרון, כפי שהוסבר בארכיטקטורה בפרק 6. בשרותות אלו הסדרה הנקסוט לרשף עוגרת דריך encoder היזיר וקטור בגודל ידוע ווראש המיציג את הסדרה המקורית, תוך התחשבות בסדר של ייברי הסדרה ובקשר ביןיהם. לאחר מכן עובר זה עוגר ב-decoder שיכל לפענוח את המידע שיש בווקטור וליחסים אותו בזורה אחרת. למשל בתרגום משפה לשפה(seq2seq – מודול של מילוי משפט בשפה אחת לווקטור מסוים ולאחר מכן מפענחו את הווקטור לשפה השנייה).

הדרך המקבילה ליצור את הווקטור ולפענחו אותו הייתה שימוש בארכיטקטורות שונות של RNN, כמו למשל רשות עמוקה מסוג LSTM או GRU המכילה ריכבי זיכרון. מודלים אלו נתקלו בבעיה בסדרות ארוכות, כיוון שהווקטור מגבל ביכולת שלו להכיל קשרים בין מספר רב של איברים. כדי להתמודד עם בעיה זו ניתן לנகוט בשיטה שונה – במקומם ליזיר וקטור במצב encoder,encoder-Decoder,Decoder הפלט לאיברי general attention (general attention) וקשרים בין איברי Decoder,Decoder,Decoder עצם (self-attention). ניקח לדוגמה תילוי בין איברי הפלט של מילוי self-attention ("How was your day"). מנגנון self-attention לשפה אחרת – במקורה זה מנגנון attention מייצר וקטור חדש עבור כל מילה בסדרת הפלט, כאשר כל ריכב בווקטור מכמת עד כמה המילה הנוכחית במצב קשורה לכל אחת מהמילים במילוי המקורי. באופן זה כל איבר בסדרת הפלט ממשקל כל אחד מאיברי סדרת הפלט. מנגנון זה נקרא attention.



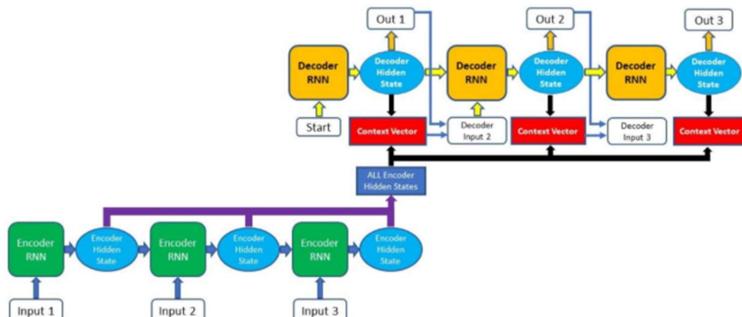
איור 8.1 מנגנון משקל – נתינת משקל לכל אחת ממלילות הקלט בהתאם לכל אחת ממלילות הפלט.

במאמר משנת 2017 שנקרא "Attention is All You Need", הוצע להשתמש ב-attention בלבד ללא רשותות מסווג או GRU LSTM, ומאמר זה פוץ לשימושים רבים במנגנון זה תוך קבלת ביצועים מעולים.

בחילק זה יוצגו הגישות המשלבות בין רשותות RNN לבין attention self-attention ו-self-position encoding. ב-attention self-attention מושג ייצוג המילויים באמצעות מילויים נסונים (position embeddings) ובקטגוריה זכר/נקבה (gender embeddings).

8.1.2 Bahdanau Attention and Luong Attention

הגישה הראשונה שהוצענה נקראת Bahdanau Attention – על שם הממציא שלה – .Dzmitry Bahdanau



איור 8.2 ארכיטקטורת Bahdanau Attention

הweeney של גישה זו היא לבנות ארכיטקטורה בה משתמשים בכל המבדים החבויים של רכיבי הזיכרון ב-*encoder*-*decoder*. מושגאת מכך ה-*decoder* מחשב את המוצא לא רק על סך ממצוי הקודמים, אלא משקל אתם יחד עם המבדים החבויים של ה-*encoder*. עבור כל אחד מאיברי סדרת הפלט מושגים alignment score בין המצב החבוי של רכיב הזיכרון הקודם בסדרת הפלט לבין כל המבדים החבויים של ה-*encoder*, וכך יוציאים שבעזרתו מחשבים את הפלט עבור האיבר הנוכחי ב-*decoder*. ביצוע הפעולה זו הוא הלב של מנגנון ה-*attention*, כיון שהוא קשור בין הפלט לפלט, ובנוסף מחשב עבור כל איבר של סדרת הפלט כמה משקל יש לתת לכל אחד מאיברי הפלט האחרים.

ביצוע פעולה זו יוצרת לכל אחד מאיברי הפלט context vector ייחודי משלו הנבנה גם מהמצוב הקודם וגם מאיברי ה-*encoder*, בשונה מארQUITECTURES הקודומות של seq2seq בסיסן לא היה ניתן ליצור מידע בעבר שימוש מהմבדים החבויים של ה-*encoder*-*decoder*. את ה-*attention*-score מוחברים למצואו של האיבר הקודם ב-*decoder*, יחד עם המצב החבוי הנוכחי, יוציאים את המצב החבוי הבא, שבעזרתו מוצאים את הפלט של האיבר הנוכחי.

באופן פורמלי, אם נסמן ב- H_e, H_d את המבדים החבויים של ה-*encoder* וה-*decoder*, ה-*alignment score* יתקבל על ידי:

$$\text{alignment score} = w_{\text{alignment}} \times \tanh(w_d H_d + w_e H_e)$$

כאשר $w_{\text{alignment}}, w_d, w_e$ הם המשקלים הנלמדים של ה-*decoder*, ה-*encoder* והחיבור ביניהם. את התוצאה מعتبرים דרך SoftMax, מכפילים ב- H_e ומקבלים את ה-*context vector*:

$$\text{context vector} = H_e \times \text{SoftMax}(\text{alignment score})$$

הווקטור המתתקבל מכיל משקל של כל אחד מאיברי הפלט ביחס לאיבר הפלט הנוכחי. את התוצאה כאמור מוחברים לפלט של האיבר הקודם, ובעזרת המצב החבוי הקודם מחשבים את המצב החבוי הנוכחי, שמננו מחליצים את הפלט של האיבר הנוכחי.

ישנו שיפור של Bahdanau attention הנקרא *Loung attention*. שני הבדלים העיקריים יש בין שני המנגנונים: חישוב alignment score מתבצע באופן שונה, ובנוסף בכל שלב לא מושגים ממצוב החבוי הקודם של ה-*decoder* כמו שהוא אלא יוצרים מצב חבוי חדש ובעזרתו מחשבים את ה-*attention score*.

8.2 Transformer

לאחר שמנגנון ה-*attention* התחליל לצבור תואוצה, המוצאה ארכיטקטורה המבוססת על *attention* בלבד ולא שום רכיב זיכרון. ארכיטקטורה זו הנקראת transformer מציעה שני אלמנטים חדשים על מנת למצוא קשרים בין איברים בסדרה מסוימת – self-attention ו-positional encoding.

8.2.1 Positional Encoding

ארQUITECTURES מבוססות RNN משתמשות ברכיבי זיכרון בשbill ליקחת בחשבון את הסדר של האיברים בסדרה. גישה אחרת לייצוג הסדר בין איברי הסדרה נקראת positional encoding, בה מושגים לכל אחד מאיברי הפלט פיסת מידע לגבי המיקום שלו בסדרה, והואופיה זו כפופה באיה כתחילף לרכיבי הזיכרון ברשותות RNN. באופן פורמלי, עבור סדרת קלט $\mathbf{x} \in \mathbb{R}^d$, מחשבים וקטורי מממד $1 \times d$ באופן הבא:

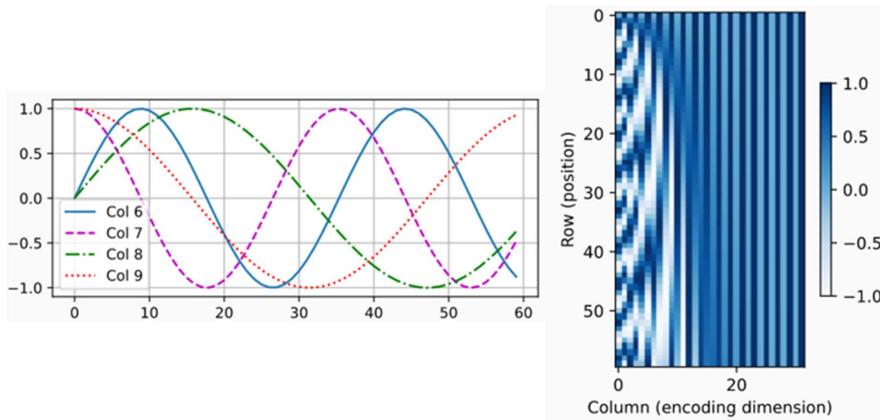
$$p_t(i) = \begin{cases} \sin(\omega_k t), & i \text{ is even} \\ \cos(\omega_k t), & i \text{ is odd} \end{cases}, \omega_k = \frac{1}{10000^{\frac{2k}{d}}} \rightarrow p_t = \begin{bmatrix} \sin(\omega_1 t) \\ \cos(\omega_1 t) \\ \vdots \\ \sin(\omega_d t) \\ \cos(\omega_d t) \end{bmatrix}_{d \times 1}$$

בדי להבין כיצד וקטורי זה מכיל ממשמעות של סדר בין דברים, נציג את הרעיון שהוא מייצג בצורה יותר פשוטה. אם נרצה לקחת רצף של מספרים וליציג אותם בצורה בינארית, נוכל לראות שככל שהBIT יש משקל גדול יותר, כך הוא משתנה בהתאם למזהה יותר, ולמעשה תדיירות שני היבט היא אינדיקציה למיקום שלו.

0 :	0 0 0 0	8 :	1 0 0 0
1 :	0 0 0 1	9 :	1 0 0 1
2 :	0 0 1 0	10 :	1 0 1 0
3 :	0 0 1 1	11 :	1 0 1 1
4 :	0 1 0 0	12 :	1 1 0 0
5 :	0 1 0 1	13 :	1 1 0 1
6 :	0 1 1 0	14 :	1 1 1 0
7 :	0 1 1 1	15 :	1 1 1 1

איור 8.3 ייצוג בנארי של מספרים. ה-MSB משתנה בתדריות היכי גבואה.

כיוון שמתusalemם במספרים שאינם בהכרח שלמים, הייצוג הבינארי של מספרים שלמים הוא יחסית בזבזני, ולכן מחייב רציפה של אותו רעיון – פונקציות טריגונומטריות עם תדריות הולכת וגדלה. זהו בעצם הווקטור \vec{k} – הוא מכיל הרבה פונקציות טריגונומטריות בעלות תדריות הולכת וקטנה, ולפי התדריות שמתואספת לכל איבר בסדרה המקורית ניתן לקבל אידמייציה עלי מיקומו.



איור 8.4 Positional encoding. דוגמא למספר פונקציות בעלות תדריות הולכת וקטנה, בהתאם לאיבראות ה-MSB. המבאה לקצוב השינוי של כל פונקציה בהתאם למיקום של האיבראות היא מייצגת – מען גרסה רציפה לקצוב שינוי הביטים בייצוג בנארי של מספרים שלמים (ימין).

ישנו יתרון נוסף שיש לשימוש בפונקציות הטריגונומטריות – עבור כל צמד פונקציות בעלות אותו תדר $\begin{bmatrix} \sin(\omega_k t) \\ \cos(\omega_k t) \end{bmatrix}$, ניתן לבצע טרנספורמציה לינארית ולאחר מכן:

$$M \cdot \begin{bmatrix} \sin(\omega_k t) \\ \cos(\omega_k t) \end{bmatrix} = \begin{bmatrix} \sin(\omega_k t + \phi) \\ \cos(\omega_k t + \phi) \end{bmatrix}, M = \begin{bmatrix} \cos(\omega_k \phi) & \sin(\omega_k \phi) \\ \cos(\omega_k t + \phi) & -\sin(\omega_k t + \phi) \end{bmatrix}$$

באופן זהה מקבלים באופן מיידי ייחוס בין כל-positions, מה שיכל לעזור בניתוח הקשרים שבין איברים שונים.

8.2.2 Self-Attention Layer

בנוסף ל-*positional encoding*, עלה הרעיון לבצע attention לא רק בין איברי הפלט, אלא גם בין איברי הקלט עצמום. הרעיון הוא לייצר ייצוג חדש של סדרת הקלט באמצעות אורך כמו הסדרה המקורית, כאשר כל איבר בסדרה החדש ייאפשר איבר בסדרה המקורית בתוספת מידע על הקשר שלו לשאר האיברים. הרעיון הכללי אומר שיש לקחת כל איבר בסדרה, ולה חשב את הדמיון שלו לאור האיברים בסדרה. איברים דומים (קרובים) בסדרה יקבלו ערכי דמיון גבוהים, ואילו איברים שונים (רחוקים) בסדרה יתייחסו ערכיהם נמוכים (ב-NLP זה יכול להיות מילימ' שסביר שייפשו בסיסיות, ובתמונה זה יכול להיות פיקסליהם דומים). דמיון בין איברים מدد על פי הקשר שיש ביניהם, והוא מחושב באמצעות מכפלה פונימית בין וקטוריו ייצוג של האיברים. כל מכפלה פונימית בין שני איברים נתונה מקדם שהוא מספר ממשי, וכך ניתן לסייע את מכפלת כל המקדמים באיברים המקוריים, ולאחר מכן חדש לאיבר המקורי

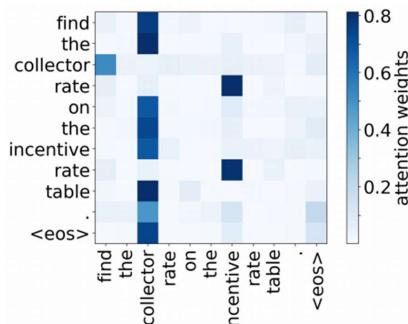
המיכל גם קשור בין האיבר הנוכח לאיברים דומים בסדרה. במקרים אחדות, ניתן להסתמך על וקטור המיכל את הקשרים של איבר מסוים בסדרה כדי ליצור מחדש את המשקף שקשרו עם שאר איברי הסדרה.

בabaon פורמלי, בשביל לחשב את self-attention self-attention יוצרים שלוש מטריצות של מקדים עבר סדרת ה cinematic. המטריצות נקראות Query, Key, Value, כאשר כל אחת מהן נוצרת על ידי הקפלה של מטריצת משקלים באמצעות attention score את self-attention score. בעדרת מטריצות אלו מחשבים את self-attention score את self-attention score.

$$\text{Attention}(\text{Query}, \text{Key}, \text{Value}) = \text{SoftMax}\left(\frac{\text{Query} \cdot \text{Key}}{\sqrt{d_k}}\right) \cdot \text{Value}$$

כדי להבין כיצד הנוסחה הזו מסייעת קשר בין איברים, נבחן כל איבר שלה בנפרד. עבור סדרת קלט x מקובלים שלוש מטריצות, כאשר כל איבר בסדרה המקורית x_i יוצר שורה בכל אחת מהמטריצות. כאשר לורוקחים את השורה $x_i = q_i$, ומכפילים אותה בכל אחת מהשורות במטריצה K , מקבלים וקטור חדש, שיכל איבר j בוקטור אונומר עד כמה יש קשר בין האיברים i, j בסדרה המקורית. ביצוע ההכפלה הזו עברו על סדרת הקלט יוצר מטריצה חדשה בה כל שורה מייצגת את הקשר בין איבר מסוים לשאר איברי הסדרה. ההכפלה הזו היא בעצם $K \cdot Q$, כאשר Q embedding-ההכפלה של מילוי k מיצגת את הקשר בין האיבר j . את התוצאות החלקיים בשורש של מודול embedding -ההכפלה כל מכפלה $q_i K$ מיצגת את הקשר בין האיבר i לאיבר j .SoftMax. ולאחר מכן מוגבלים מטריצה של מספרים בטוחה לשומר על יציבות הגרדיינט, ולאחר מכן מנורמלים על ידי SoftMax . באופן זה מוגבלים מטריצה של שני איברים בסדרה המוקורית. בסמן כל איבר במטריצה b_{ij} , ונוכל לקבל [0, 1], המציגים כאמור את הקשר בין כל שני איברים בסדרה המוקורית. ואנו שירוט על ידי הנוסחה:

$$w_{ij} = \text{SoftMax}\left(\frac{\mathbf{q}_i^T \mathbf{k}_j}{\sqrt{d_k}}\right) = \frac{\exp\left(\frac{\mathbf{q}_i^T \mathbf{k}_j}{\sqrt{d_k}}\right)}{\sum_{s=1}^n \exp\left(\frac{\mathbf{q}_i^T \mathbf{k}_s}{\sqrt{d_k}}\right)}$$

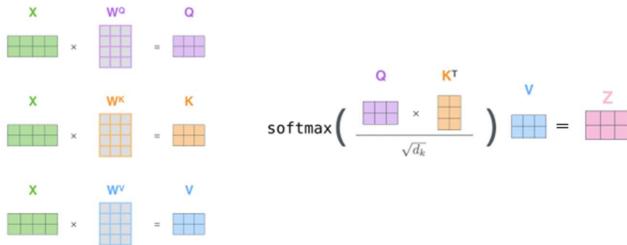


איור 8.5 מתריצת משקלים של המפט"ה. Find the collector rate on the incentive rate table."Find the collector rate on the incentive rate table".

כעת בעזרה משקלים אלו בונים ייצוג חדש לסדרה המקורית, על ידי הכפלתם בוקטור ψ :

$$z_i = \sum_{j=1}^n w_{ij} v_j = \frac{\sum_{j=1}^n \exp(q_i^T k_j)}{\sum_{s=1}^n \exp(q_i^T k_s)} v_j$$

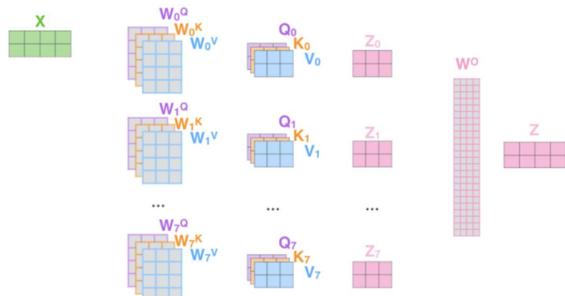
הסדרה המתקבלת Z היא למעשה ייצוג חדש של הסדרה המקורית, כאשר כל איבר Z_t מיצג איבר בסדרה המקורית. אט הסדרה המתקבלת ניתן להעיבר-*decoder* המכיל שכבות נוספות, ובכך לבצע מלמעלה מיפוי משימות, כפי שיאסביר בהמשך.



אייר 6 ביצוע Self-attention score – ייצרת מטריצות (Query, Key, Value) (שמאלו) וחישוב ה-*score* (ימין).

8.2.3 Multi Head Attention

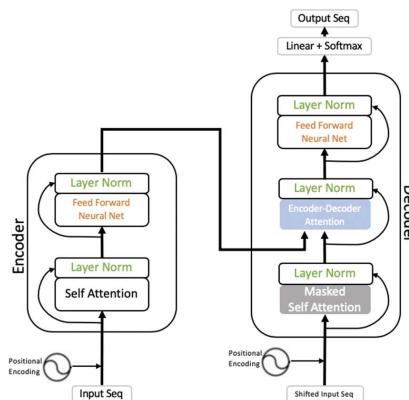
ניתן להשתמש במנגנון self-attention מספר פעמים במקביל. כל פעם מקבלים שלוש מטריצות (Q_r, K_r, V_r), מחשבים את הייצוגים החדשניים של איברי הסדרה (attention score). כל מנגנון זהה נקרא head, ויחbero במקביל של כמה heads נקרא Multi-head attention. באופן זהה לכל איבר כניסה x_i יש כמה ייצוגים שונים z_i , אותם ניתן להכפיל במטריצה משקלים W_o ולקבל את הייצוג המשוקל של אותו איבר באמצעות *attention heads* רבים.



אייר 7 Self-attention with 8 heads 8.7

8.2.4 Transformer End to End

בעזרת מנגנוני multi head attention ו-positional encoding ניתן לבנות Transformer – ארכיטקטורה עבור סדרות המבוססת רק על attention ללא רכבי זיכרון.



אייר 8 Transformer 8.8

כפי שניתן לראות באירוע transformer – מרכיב משני חלקים – encoder ו-decoder. ה-encoder מקבל סדרה מוצפנית x (לרוב אחריו שعبירה embedding מסוימת) ובמציע עלייה positional encoding. לאחר מכן המסדרה עוברת דרך self, attention התוצאה (x) + x . על תוצאה זו מבצעים residual block של normalization, fully connected residual block בפקט. לאחר מכן שבסוף, המכילشبת encoder ה-encoder מוציא הפלוטן לAYER.

decoder בינו ביצורה מואוד דומה, עם שני הבדלים עיקריים: הקלט שלו הוא איברי הפלט שהיו עד כה, ובו סוף יש בתחלת ה-decoder שכבה של Masked self-attention. שכבה זו מקללת את כל איברי הפלט שהיו עד כה, ומטרת ה-decoder היא ללמידה בעצמו מה האיבר הבא של הפלט. בשלב הראשון ה-decoder מבצע self-attention על איבורי הפלט שקדמה לו, וכך לא ישים חידושים סלולים. הורילס גם את היחס בין איבורי הפלט זו.

לאחר השכבה הראשונה יש שכבת multi head attention נוספת הנקראת Encoder-Decoder Attention. בשכבה זו יש שילוב של encoder וה-decoder: המאפייניות **Key**, **Value**encoder מה-**Query** וה-decoder. כעת כשמבצעים את המכפלה $K \cdot Q$, לא מחפשים דמיון בין איברים של אותה סדרה אלא בין האיברים של סדרת הפלט (encoder) לבין איברים סדרת הפלט (byizing שלם לאחר שכבת ה-**masked**). בשלב זה זומנה מודול attention המוקורי, רק שה**Query**ים שהתקבלו לא נעדزو ברכיבי **Z**ircor. כאמור, המכפלה $K \cdot Q$ מייצרת מטריצת משקלים לכל איבר בה אומר מה היחס בין איבר בסדרה המקורית לבין איבר בסדרת הפלט. את המטריצה זו מכפלים ב-**V** וצר מתגבל איבר הפלט. **SoftMax** FC.

הנתק דוגמא ממאמר שנקרא **DETR** המראה כיצד ניתן להשתמש ב-**transformer** בשביל ציהוי אובייקטים בתמונה. בשלב הראשון לוקחים כל פיקסל בתמונה ומשווים אותו לשאר הפיקסלים (זהו בעצם המכפלת $K \cdot Q$). באופן זה ניתן למצוא איזורים דומים ושונים בתמונה, כאשר דמיון ושוין זה לאו דווקא פיקסלים עם ערבים קרובים, אלא זה יכול להיווכח למשל שניים בפניהם של אדם. לאחר מכן מיצרים ייצוג חדש בתמונה, באמצעות משקלים והופכים אותו ל-**B**. בשלב זה למעשה מושך ליצוע ציהוי של אובייקטים, בלי לדעת מה הם אוטם אובייקטים. בשביל לצUGHSIIG **Query** ליל אובייקט שזהה, מعتبرים את היצוג החדש של התמונה ב-**decoder**, כאשר ה-**decoder** שמנכזים זה כל מי-ליילים אפשריים, ומוחפשם מבין כל ה-**Query** את הפלט של ה-**decoder** שמציל יוצר תמונה שהכי דומה ל-

אם למשל יש תמונה גודלה ויש אזכור מסוים בו יש חתול, אז ה-encoder מוצא איפה החתול בתמונה, וה-decoder משווה את האזכור הרה כל מני חיות אפשריות. כל **Query** שללא היה חתול, המכפלה K - Q היה קרובה ל-0, וה-decoder יזהה שה-Query הנוכחי לא תואם לאובייקט שゾהה. אך כאשר ה-Query היה חתול, אז כיוון ש- $K = Q$, הדומים אחד לשני, המכפלה K - Q תביא לכך שהשיזוג החדש $\sum_{j=1}^n w_j t_j = z_i$ כי היה דומה לחתול. ציוג זה עבור בשכבת FC, ולאחר מכן SoftMax יוציא את התמונה הרצויה כחטול.

8.2.5 Transformer Applications

ה-Transformer מושם במקומות רבים. מנגנון state-of-the-art בפיזיקה, למשל, הוא היזוה השראה להמן יישומים הנעשנים על attention transformer. מלבד הרמה הגבוהה של הביצועים, תהליכי האימון של transformer הראים הוא הרבה יותר מהיר מרשותות קובולוציה או רשותות קווטיביות. כמו מודולים אחרים, גם transformer יתאפשר לבצע transfer learning, כלומר לחקות变压器 שואומן על משימה מסוימת, ולהתאים אותו למינימנה חדשה המשימה המקורית. בזכות לא כל היישומים משתמשים בכל ה-Transformer, אלא בהתאם לשימושו לוחצים חלקים מסוימים שלו ובונים עבורה משימה מסוימת. נביא מספר דוגמאות:

Machine Translation – תרגום משפטים בין שפות שונות הוא שימוש טריוויאלי של ה-transformer המלא. המשמש היא לחקת משפט ולהוציא משפט בשפה אחרת, וזה נעשה בערך י"צ המשפט המקורי באמצעות חישוב self-attention ולآخر מכ המרת בערת Encoder-Decoder Attention לשפה אחרת.

במהלך כל פעם באupon רנדומלי עושים masking על מילים מסוימות, ומטרת המודול הוא לחזות את המילים החסרות. השפה היא פונקציה המקבלת קלט טקסט ומחזירה את ההתקפות למליה הבאה על פי כל המילים במילון. השימוש היכי מօר ויאנושואטיבי של מודול שפה הוא השלהmA אוטומטית, שמצויה את המילה או המילים היכי סברות ביבינן מה שהמשתמש הקליד עד כה. כאשר מוצאים self-attention על משפטים, למעשה מקבלים "יצוגים חדשים שלהם יחד עם הקשרים בין המילים השונים. לכן encoder-in-encoder יכול ליצור מודול שפה, אם מאמנים אותו בצוואר מתאימה. המפתחים של BERT בו encoder מקבל כל מין משפטים בשני כיוונים – גם מהתחלה לסיום וגם מהסוף להתחלה, ורק היצוגים שנלמדו קיבלו קונטקטס שלם יותר. בנוסף, הם אימנו את המודול על משפטים בהם כל פעם באupon רנדומלי עושים masking על מילים מסוימות, ומטרת המודול הוא לחזות את המילים החסרות.

מהי המילה הבאה באמצעות **decoder** בלבד. מכיוון משפט קטוע וודקים את ההאמה שלן למשפט הנוכחי, והמילה שהכי מתאימה נבחרת להיות המילה הבאה. המשפט הקטוע הוא למשה **the**, **Key**-ו**Query**-שנכוו הוא כל פעם מילה אחרת במילון, וכך בעדות **attention** בוchnim איזה יתאים בצורה הטובה ביותר **Key**-ל**Query**.

References

<https://arxiv.org/abs/1409.0473>

<https://arxiv.org/abs/1706.03762>

<https://towardsdatascience.com/day-1-2-attention-seq2seq-models-65df3f49e263>

<https://towardsdatascience.com/transformer-attention-is-all-you-need-1e455701fdd9>

<https://arxiv.org/abs/1810.04805>

9. Computer Vision

9.1 Object Detection

זיהוי אובייקטים היא משימה מאוד נפוצה בעולם של ראייה ממוחשבת, וש לה המון יישומים מגוונים. ניקח למשל מכונות אוטונומיות, שבכל רגע צריכה להזיהות את האובייקטים שבסביבה ולקבל תמצאות מובן עצמאי על המתרחש, או לחילופין מצלמה שהולפּן בידי שידועת להזיהות פנים של בנאדם בכדי לבצע עליהם פעולה או לתקן רעש רקע, ועוד המון יישומים נוספים בכל מיני תחומיים.

בשלב ראשון יש להגדיר באופן מדויק את המשימה – מה הכוונה להזיהות אובייקט בתמונה? יש כמה רמות שנות של זהה. המשימה הקלאנית של סיווג (classification) מבקשת שיש אובייקט ייחד בתמונה, והמטרה היא לסווג אותו בחרורה נכונה, ככלור לקבוע מה class שלו. המשימה יותר מתקדמת היא לא לומר לנו איזה אובייקט נמצא בתמונה, אלא גם לומר איפה בדיקן הוא נמצא. אם פה יש שתי רמות – ניתן לסמן את האזור בו הוא נמצא בערבה מלבד (bounding box) שמקיף את השולדים שלו, וכן ניתן לסווג כל פיקסל בפני עצמו האם הוא שייך לאובייקט או לרקע. זההו מושג הראשוני ונכנס תחת התחום של **Object Detection** ואילו זיהוי סמנטי של הפיקסלים נכנס תחת התחום של **Segmentation**.

ניתן להכליל את משימת הזיהוי לכמה-object multiple object. לעומת זאת, יש מספר לא ידוע של אובייקטים בתמונה, והמטרה היא למצוא הין הם נמצאים ולסווג כל אחד מהם לכ-class המתאים. בעצם משימה זו מרכיבת משתת'תית משימות – מציאת המיקום של האובייקט (Localization) וסיווג האובייקט לכ-class classification (Classification). נשים לב שעבור משימת Object Detection ובעבור כל אחד מהם להתאים class. במשימת Segmentation המודל לספק מספר אובייקטים יכול להיעשות בשתי דרכים: (1) Semantic Segmentation – שיר כל פיקסל לכ-class מסוים ללא הבחנה בין פיקסליהם השימושיים לאובייקטים שונים בעלי את samme class. במרקחה זה אם ייינו שיכילם בתמונה, אז במפת הסגמנטציה נראה הרבה פיקסלים המשווים לכ-class של כל, אך לא יוכל להבחין בין הכלבים השוניים, ואפילו לא נדע שיש יותר מכלב אחד. (2) Instance Segmentation – שיר כל פיקסל לכ-class מסוים תוך הבחנה בין פיקסליהם המשתויים לכ-class ספציפי אך שיכולים לאובייקטים נפרדים.

בפרק זה נדון במשימת Object Detection ובפרק הבא ב-**Instance Segmentation**.

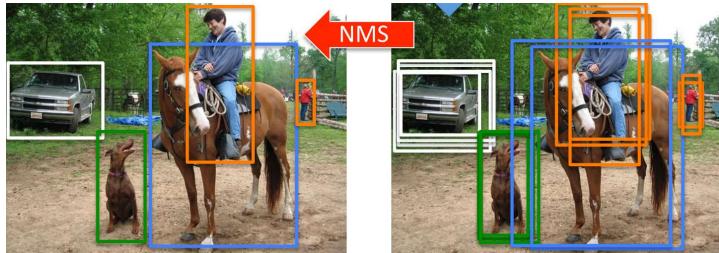
9.1.2 You Only Look Once (YOLO)

עד שנת 2016 כל הгалמים (detectors) המבוססים על למידה عمוקה (כמו למשל R-CNN family) היו גלאים דו-שלבים – השלב הראשון יצר אלף הצעות (proposals) למוגרות מבניות החשודות כמכליות אובייקטים, ולאחר מכן בזו אחר זו השלב השני דיק את המוגרות וביצע סיווג לאובייקטים המוכלים בה. ארכיטקטורת YOLO, הנגזרת מרישי התיבות של ONLY LOOK ONCE ONLY, הוצאה על ידי ג'וזף דרמן ב-2016, והייתה הגלאי הראשון שמודרך משלב ייחד, ובו הרשת מבאת את המוגרות ומוסוגת את האובייקטים שבתוכן בבת אחת. בנוסף, ארכיטקטורת YOLO מתאפיינת במעטפת פרטיטרים וכמוות פעולות אРИטמיות. היא אמנת משלמת על המבנה הרכה והאלגנטני שללה בדיק נורוות, אך המהירות גבוההה (שנובעת בעקבות המהדר האלואץ לעיבוד מסגרת יחידה בשלב השני בכל מעבר של הולאה) הפקה את גישת השלב היחידי לאטרקטיבית מאוד, במיוחד למחשבים קלים כדוגמת מכשירי mobile. בעקבות עבודה זו פותחו גלמים רבים על בסיס שלב יחיד, כולל גרסאות מתקדמות יותר של YOLO (הगרסה המתקדמת ביותר כיוון היא 5).

NMS (Non-Maximum Suppression)

כמעט כל אלגוריתם של זיהוי אובייקטים מייצר מספר רב של מוגרות חדשות, כאשר רובן מיותרות ויש צורך לדלול את מספן. הסיבה לייצור מספר גדול של מוגרות נובע מאופי פועלות הгалמים. בזכות התכונות הלקלאליות של פועלות הקונבולוציה, מפת הפייצרים במקומות הgalaxy נינטנת לתיאור כמטריצת משbezחות כאשר כל משbezח במקומות של הרבה פיקסלים בתמונה המקורית. רוב הгалמים פועלים בשיטת עוגנים (anchors), כאשר כל משbezח במקומות הgalaxy מנבאת מספר קבוע של מוגרות שעשוויות להכיל אובייקט (למשל, ב-YOLOv2 המספר הוא 5, ובגרסאות המתקדמות יותר המספר הוא 3). השיטה זו יוצרת אלפי מוגרות שרק מועtot מהן הן ממשמעותיות. בנוסף – יש ריבוי של מוגרות דומות סבביה כל אובייקט (למשל YOLOv3 מנבאת יותר מ-7000 מוגרות לכל תמונה). אחת הדריכים הפופולריות לסנן את אלף המוגרות ולהשייר רק את המשמעותיות נקראת NMS. בשיטה זו מתבצעת השוואה בין זוגות של קופסאות מאותה המחלקה (למשל – חתול), ובמידה שיש ביןיהם חיפפה גבוהה – מוחקים את המוגרת בעלת הווהות הנמוכה יותר ונשארים רק עם המוגרת בעלת רמת הווהות הגבוהה. שיטה זו בזבזנית

בחישוב (סיבוכיות פרופורציונלית לריבוע מספר המסגרות) ואינה חלק מהמודל המתאים, אך עם זאת היא נינה אינטואיטיבית יחסית למימוש, ומשום כך נמצאת בשימוש נפוץ בגלאים, כולל ב-YOLO.

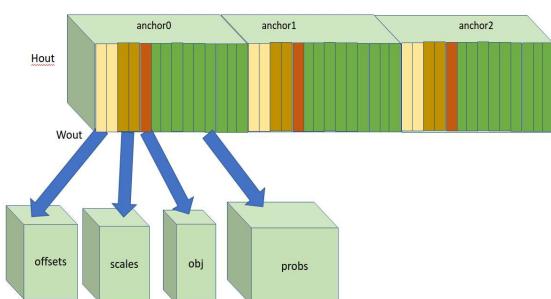


אייר 9.1 אלגוריתם NMS.

YOLO Head

כמו רוב הгалאים, YOLO הינו מבוסס-עוגנים (anchor-based), שהין תיבות מלכניות קבועות ושונות זו מזו בוצרותן. לכל עוגן מוקצתה מוקטעה של פיצרים במפת המוצא של הרשת וכל הינוויים במקטעה זהה מקודדים סטיות (offsets) (offsets) ביחס למרכז העוגן. כפי שנניתן לראות באייר 9.2, הפיצרים של כל תא מרחבי במפת המוצא מוחולקים על פי העוגנים (שלושה עוגנים במרקחה זהה).

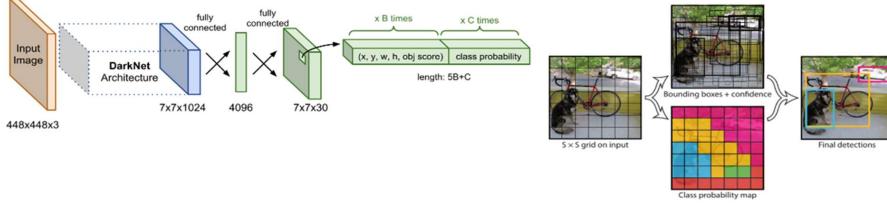
נביו הוא מסגרת שמרכזה נמצא בנקודה כלשהי בשטח התא (השקלול לריבוע של מספר פיקסלים בתמונה המקורית). ההסתה המדעית של מרכז המסגרת ביחס לתא ניתנת על ידי שני הפיצרים הראשונים ברצף. לוג הממדים של הקופסה (ביחס למרכז העוגן) ניתן באמצעות שני הפיצרים הבאים ברצף. הפיצר החמישי לומד את מידת האובייקט (objectness), כפי שהוחזרה לעיל, שאר הפיצרים ברצף של העוגן הנ"ל הם הסתבותיו המותנות לכל מחלקה (אם אוסף הנתונים מייל 80 מחלקות, יהיו 80 פיצרים כאלה). על מנת לקבל רמת ודוות סופית, יש להכפיל את מדד ה-objectness במדד הסתבותו המותנה לכל מחלקה.



אייר 9.2 ראש YOLO.

YOLOv1

מודל YOLO מבוסיס על גרסאות Darknet Backbone והקראות detection המתקבלות מtower התמונה, ומרתף המקלט את הפיצרים האלה ומתאים לייצר מהם נינויים למסגרות סביב אובייקטים. המודל מחלק את התמונה לרשת בעלת $S \times S$ משבצות, אשר כל משבצת מנובאת N מסגרות של אובייקטים בשיטת העוגנים, כאמור לעיל. כל נביו כולל את מספר ערכי: הסתה הקאוורידנטות y, x , של מרכז המסגרת ביחס למשבצת, הגובה והרוחב של המסגרת, ורמת objectness-העוגן, כדי שהוחזרה לעיל. בנוסף, כל מגרת מבצעת גם סיג, כמו רוחב המסגרת ורמת הווודאות של השתייכות האובייקט לכל אחת מהמחלקות האפשרויות. החידוש באלויגרים נועד בעובדה שחייבי המסגרות וסיוון לאובייקטים נעשה במקביל, ולא באופן דו-שלבי. הרעיון הוא להתייחס לסוג האובייקט כעד פיצר שהרשת מנסה לחזות בנוסף למיוקם וגודל של המסגרת.



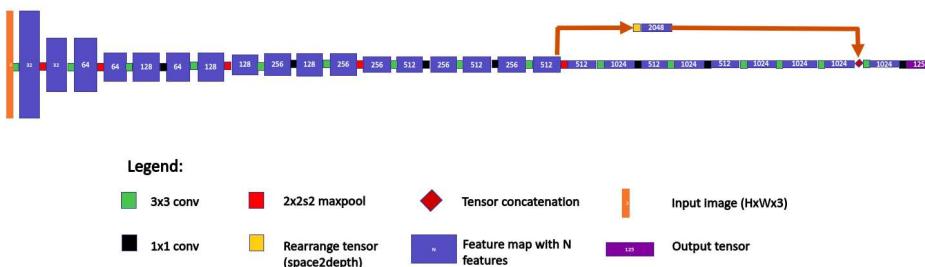
איור 9.3 ארכיטקטורת YOLO.

הגרסה הראשתית של ארכיטקטורת YOLO כללה שכבה fully connected שהוסרה בגרסאות הבאות. בנסוף, פונקציית LOSS וודר התכוונת של המוגרות בМОץ הרשות השתנו, אבל הרעיון נותר זהה.

YOLOv2

גרסה זו משתמשת ברשת הנקראת Backbone Darknet19 ובها 19 קובולוציות. המודל כולל 22 קובולוציות (מלבד 5 שכבות MAXPOOL המקבילות על מנת למצוא את הפיצרים), ועוד מסלול עוקף בסופו הרשות המחזק את יכולת היעבוד. הכותב של המאמר המקורי, רדמן, נהג לפתח גם גרסאות "Tiny" לכל מודל. הוויאנט נמוכם יותר אך הוא מהיר מאד (207 תמונות לשנייה לעומת 67 של מודל YOLOv2, על מעבד X-Titan).

משמעותי ש-YOLOv2 הוא המודל הראשון שאומן על תמונות במדדים משתנים, תהליך המשפר את דיק המודל.



איור 9.4 ארכיטקטורת YOLOv2.

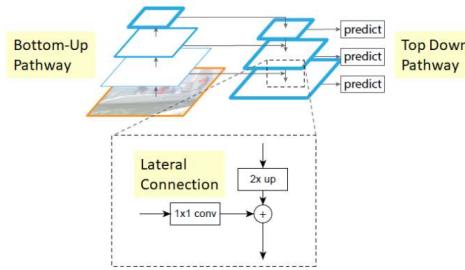
YOLOv3

כל דור נוסף של YOLO הציג חידושים ארכיטקטוניים שהגדילו את מורכבות החישוב ואודל המודל ושיפורו את ביצועיו. גרסתו מספר 3 מבוססת על רשת Backbone אגדולה בהרבה שנקראת Darknet53 ובה, כפי שעה לה, 53 קובולוציות. כמו כן רשת מכילה צוואר של ארכיטקטורת Feature Pyramid Network (FPN) בעלת שלושה ראשים (גיליון, כשר כל אחד מהם הוא בעל חיזוקה שונה ($38 \times 19 \times 38$ ו- $76 \times 76 \times 38$)).

אררכיטקטורת FPN היא תוספה בקצתה-h Backbone, אשר מגדילה חזרה באופן הדרגי את מפת הפיצרים. תוספה זו הנעודה ליצור מסלול Top Down במקביל למסלול-h Backbone Feed Forward-bottom Up, שבוני למעשה בצורת מפת הפיצרים חולפת וקטנה. בעודת השימוש ברשת-h FPN המודול לומד לצלץ את המטיב משני עולמות: הוא משתמש במידع שטחוני במפת הפיצרים הגדולה, שהיא אמנים פחות מועבדת אך בעלת פיצרים ברוחזות מרחבית גבוהה, ובנוסף הוא מנצל גם את המידע ממפת הפיצרים הקטנה, שהיא אמנים בעלת פיצרים ברוחזות מרחבית נמוכה, אך עם זאת היא מועבדת יותר.

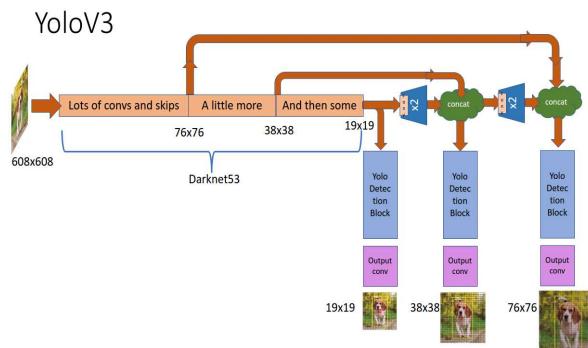
לאחר כל הגדלה של מפת הפיצרים מתבצע חיבור בין התוצאה לבין מפת פיצרים קדימה יותר במדדים זרים (מתוך Backbone). זאת בדומה לחיבורם העקיפים ברשת ResNet המסייעים להתכנסות האימון. השכבות השונות של

רשת FPN מאפשרת לאלא' למצוא מיקום מדויק יותר של האובייקט בחרזולzieות השונות, מה שמעניק לרשת זו את יכולת להבחן אובייקטים קטנים בתמונה גדולה.



איור 9.5 ארכיטקטורת FPN, המשלבת מסלול top down לאחר ה-up bottom.

ראש המודל של YOLOv3 יש מספר ענפי detection, אשר כל אחד מהם פועל על מפת פיצרים בחרזולzieה שונה ובאופן טבעי מתמחה בגלי אובייקטים בגודל שונה (הענף בעל הרזולzieה הנמוכה מתמחה בגלי אובייקטים גדולים, והענף בעל הרזולzieה הגבוהה מתמחה בגלי אובייקטים קטנים).



איור 9.6 ארכיטקטורת YOLOv3.

YOLOv4

רשת YOLOv4 היא בעלת ראש זהה לזה של שתי הגרסאות הקודמות, אך ה-Backbone שונה ומורכב יותר. הוא נקרא CSPDarknet53 כאשר CSPNET היא קיצור של Cross-Stage Partial Network. רשת זו מפעילה מיפוי פיצרים לוטובת קונבולולציה בחלקים ואיחוד מחדש. פיצול זה אפשרי, מכיוון שניתן במאמר המקורי, חילוח טוב יותר של הגדיאנטים בשלב האימון. בנוסף, נעשה ברשת זו שימוש בפונקציית Akribititska הנקראת Mish (ולא Leaky ReLU).

כמו בגרסאות הקודמות – החל מגרסה זו התכורות אין של היוצר המקורי גייזף רדמן, שהחליט לפרוש מחקר ראייה אקדמייתית מעניינת – החל מגרסה זו התכורות אין של היוצר המקורי גייזף רדמן, שהחליט לפרוש מחקר ראייה ממוחשבת בגלל שיקולים אתיים של שימושים צבאים או שימושים הפגיים בפרטיות.

YOLOv5

רשת YOLOv5 מוסיפה עוד שכליילים על רשת ה-Backbone על פני זו של הדור הקודם, ומציגת אופרטור חד המשורט פיקסלים סמכים בתמונות בממד הפיצרים. אופרטור זה דואג לכך שהכיסוי לששת היא לא עמוק 3 פיצרים ממוקבל (RGB), אלא 12 פיצרים, תוך הקטנת הממד המרחב. באופן זה הרשת מתאמת לעיבוד תמונות בחרזולzieה גבוהה, ואף לזרחות אובייקטים גדולים בקהלות רבה יותר, שכן שדה התמך (receptive field) של הקונבולולציות מכל מידע משטח תמונה גדול יותר.

9.1.4 Spatial Pyramid Pooling (SPP-net)

Spatial Pyramid Pooling (SPP) הינה שכבה pooling ברשת ניורונית, שמטרתה להסיר את האילוץ של רשותה הקונבולוציה, הדורש שכבה הכנסה לרשת תכל תמונה בגודל קבוע (כמו למשל רשת VGG, המקבלת רק תמונות בגודל 224×224).

קיימים קיימים מכשירי צילום רבים – החל מצלמות יידוט, מקצועיות, מצלמות אבטחה ואף מצלמות טלפונים סולריים ורפואיים. מצלמות שונות עושות שימוש להזיהוי עצם תמונות בגודלים שונים, מגוון סיבות (למשל איכות התמונה או מטרת המכשיף). אם נרצה לבצע סיוג באוטומטית ניורונים לכל אותן תמונות, נאלץ לבצע שלב נוסף בתחילת הדרך – התאמת התמונה בכניסה לרשת.

נשים לב כי על מנת להתאים תמונה כלשהי לגודל מסוים, לחוב יוציא חיתוך (crop), או שניי גודל (resize/wrap). פעולות אלה עלולות לגרום בדיחיה עקב שניי היחס בתמונה (מתיחה/כיווץ), החסירה של פרטיהם או שילוב של השניים. מוטיבציה זו היא שהוביילה לשימוש בטכניקת SPP.

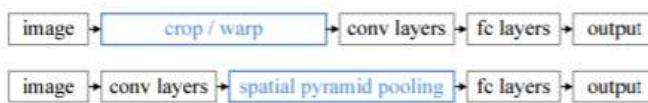
ניתן לראות דוגמא לשינוי גודל באמצעות מתיחה ולחיתוך באירוע הבא:



איור 9.7: מימין - שינוי פרופורציה. משמאל – חיתוך.

למעשה, הדרישה לתמונה קולט בגודל קבוע בכניסה לרשותה אלה אינה הכרחית, שכן שכבות הקונבולוציה יכולות לחלק מאפייני קולט (feature maps) בכל גודל. לעומת זאת, שכבות ה-FC בעומק הרשת הן השכבות שדורשות בכניסה להן קולט בגודל קבוע.

עתה, לאחר שהוביילה המוטיבטיבית לייצרת רשת זו, ניתן להבין את אופן פועלתה. על בסיס שכבת ה-SPP מתחבessa רשת CNN. החידוש במבנה הרשת, הוא שבמקומם שכבות ה-FC (לפני שכבות הקונבולוציה) תבצע התאמת תמונה הקולט לגודל הנדרש ברשota. ההתקאה המבוצעת בשכבה SPP תתבצע לאחר מציאת המאפיינים בשכבות הקונבולוציה, אך לפני שכבת ה-FC, כמפורט באירוע הבא:

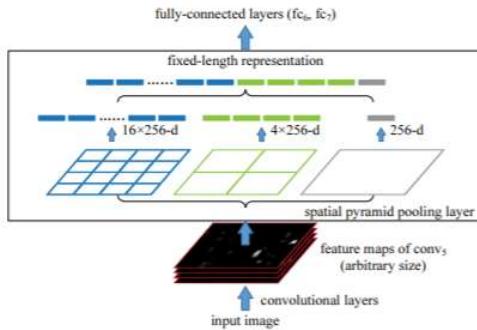


איור 9.8 – מבנה של רשת CNN קלאסית (למעלה) לעומת מבנה של רשת SPP-Net (למטה).

הרעישן מהחורי שכבת SPP הוא חילוקה של הפלט של שכבות הקונבולוציה, ביצוע max-pooling בכל חלק ושרשור של התוצאות לווקטור שגודלו אחד. במילים אחרות, עבור כל ה-*feature maps* המתפקידים לאחר שכבות הקונבולוציה, מיצרים שליטה וקטורים לכל *feature*.

- ווקטור בגודל 1 המתפרק באמצעות ביצוע max-pooling על כל הערכים באותו ה-*feature*.
- חילוקה של ה-*feature* ל-4-תת-חלקים (2×2) וביצוע max-pooling בכל חלק מתוכם. מתפרק ווקטור בגודל 4.
- חילוקה של ה-*feature* ל-16 תת-חלקים (4×4) וביצוע max-pooling בכל חלק מתוכם. מתפרק ווקטור בגודל 16.
- שרשור כל הווקטורים יחד לוקטור בעל 21 ערכי.
- בסיום התהליך, תתקבל שכבה בייזוג אחד בגודל 21, בהכפלה במספר ה-*features* שהתקבלו משכבות הקונבולוציה. שכבה זו "מחליפה" את שכבת pooling המומוקמת לאחר שכבת הקונבולוציה الأخيرة ותוצרה הוא הקולט לשכבת ה-FC.

ניתן לראות את מבנה הרשת הבא, בו המתקבלו 256 :features



אior 9.9: ארכיטקטורת SPP-Net

9.2 Segmentation

אחד האתגרים הקיימים בעולם הריאי המוחשכט הוא ציהוי אובייקטים בתמונה והבנת המתרחש בה. אחת הטכניקות הקלואיסטיות להתחמזרות עם שימושה זה הינה ביצוע סגמנטציה, כלומר, התאמת label לאלמנט מסוים בתמונה. בתקהיל הסוגנטרי מוצגים חלוקה/ביזול בין עצמים שונים בתמונה המוצלמת באמצעות סיווג ברמה גבוהה. ככלمر כל פיקסל בתמונה יסגור וישיר למולטלה מסויימת.

ישנים שימושים מגוונים באlgorigamim של סגמנטציה – הפדרה של עצמים מסוימים מהruk שמאחוריים, מציאת קשרים בין עצמים ועוד. לדוגמא, תוכנות של שיחות וועידה, כמו zoom, skype, teams וכו', מאפשרות בחירת רקעים שונים עבור המשטח, כאשר מלבד הרקע הבוחר רק הגוף של המשטח מוגז בוועידה. הפדרת גוף אדם מהruk והוסטמאות רעка אחר מtbodyות באמצעות אלגוריתמים של סגמנטציה. דוגמא נוספת – ניתן לוחות בתמונה אדם, כלב, ובוינהם צוינה, ומכך ניתן להסיק שתוכן התמונה הוא אדם מחזיק כלב בעדרת רצעה. במקרה זה, הסגמנטציה מוגדרת למצוא קשר בין עצמים ולהבין את המתרחש.

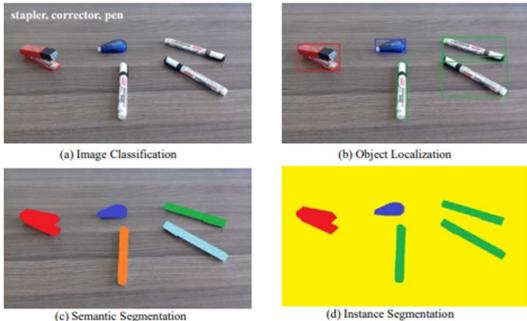
9.2.1 Semantic Segmentation vs. Instance Segmentation

ק"י מים שני סוגים עיקריים של סגמנטציה:

שייר. למשל, פיקסל יכול להיות משוייך לכל רכב, בן אונש, מבנה וכו'.

Instance segmentation (חלוקת מופעויות) - חלוקה של פיקסל בתמונה למושג של אותה מחלקה אליה העצם אותו הוא מייצג שיר. בתרמונה בה מופיעים מספר כל רכב, תבצע חלוקה של כל פיקסל לאיזה כל רכב אותו פיקסל מייצג - מכונית 1, מכונית 2, אופנוע 1, אופנוע 2 וכו'.

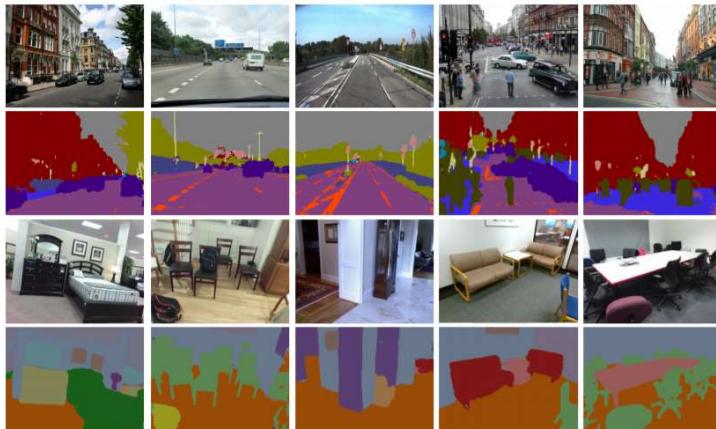
ההבדל העיקרי בין שני סוגים אלו הוא ברמת עומק המיפוי של פיקסל - המיפוי עשו לסוג את הפיקסל למחלקה כלשהיא, או לעצם ספציפי בתמונה. עומק המיפוי משליך גם על עולות המיפוי. החלטקה הסמנתית מבוצעת ישרות, בעוד שחלוקת המיפוי דורשת במעט ביצוע של זיהוי אובייקטים כדי לՏווג מופעים שונים של המחלקות.



איור 9.7 משימות שונות תחת התחום של Computer Vision בהבדל שבין Semantic segmentation (ההתאמה כל פיקסל למחילה מסוימת) לבין Instance segmentation (ההתאמה כל פיקסל למכלול של מחלקה מסוימת).

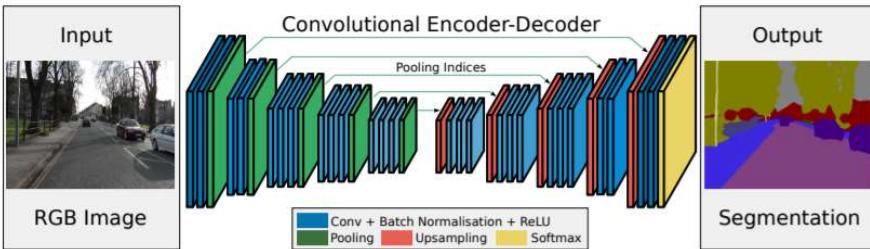
9.2.2 SegNet neural network

רשת SegNet הינה רשת קוגנולואיה عمוקה, שטרתה לבצע חלוקה סמנטית (Semantic segmentation) לתמונה הקלט. בתחילת הרשת פותחה להבנה של תכונות חז' (למשל כבש עם מכניות ובצדדים בתם והולci רגלי) ותכונות פנים (למשל חדר עם מיטה וכייסאות). הרשת נבנתה מטרה להיות יעילה בהיבטי זיכרון וזמן חישוב, תוך שמירה על דיק משען.



איור 9.7 סיווג סמנטני בעזרת רשת SegNet עבור תמונות פנים וחוץ.

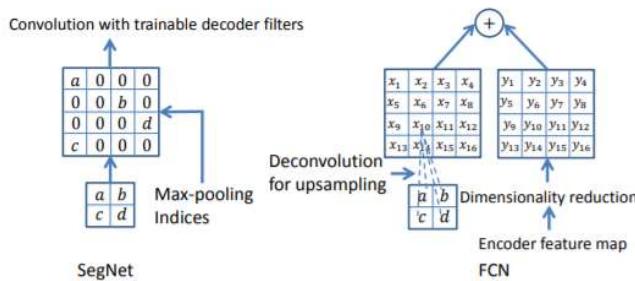
הרשת בניית ארכיטקטורת מקודד-מפענה (encoder-decoder). המקודד מורכב מ-13 השכבות הראשונות של רשת VGG16, ומטרתו לחשיך את מפת המאפיינים (feature maps). לכל שכבה קידוד יש שכבת פיענוח תואמת ומcean שגם רשת המפענה מכילה 13 שכבות. מטרת המפענה היא לבצע k-sampling ולייצר תמונות מאפיינים בגודל המקורי. את מזאת המפענה מעבירים דרך מסוג SoftMax, המתאים את המחלקה בעלת ההסתברות הגבוהה ביותר לכל פיקסל בנפרד.



איור 9.8 ארכיטקטורה רשת SegNet.

המקרה מרכיב שכבות קובולוציה, אחריהן שכבות נרמול (batch normalization) ושכבות אקטיבציה מסווג ReLU. לאחר שכבות אלו, ישנה שכבת max-pooling המבצעת max-pooling בעילת גודל חלון 2×2 ומרוחק (stride) 2. בגודל 2.

החידוש ברשת זו הוא אופן הפעולה של המפענה. בשונה מרשתות אחרות, בהן תהליכי ה-sampling נערך לאחר הפעוננו, כמתואר באירוע הבא, העזין ברשת זו הוא שמיירת מיקומי ערכי המקיים הנבחרים מכל רביעייה. רק הערכים שנבחרו כמקסימום ישוחזרו בתהליך ואילו הערכים יתאפסו.



איור 9.9 שכבת פענוח ברשת SegNet לעומת רשת FCN.

ארQUITקטורה זו מביאה את הרשות לביצועים טוביים בהיבטי זמן חישוב, על חשבן פגעה מסוימת בדיק הרשת. למראות זאת, ביצועי הרשות מתואימים לשימושים פרקטיים והפגעה בדיקת קטנה מודול.

בדומה למקודם, לאחר כל שכבת s-sampling בפעולת, יופיעו שכבות קובולוציה, שכבות נרמול ושכבות אקטיבציה מסוג ReLU. את מזיאת המפענה מעבירים דרך שכבת SoftMax המבצעת סיווג ברמת הפיקסל. מזיא הרשת, שהוא גם מזיא שכבת ה-SoftMax, הינו מטריצת הסתברותית, כאשר עבור כל פיקסל יש וקטור באורך K, כאשר K הוא מספר המחלקות לסיווג. כמובן שהסתוויג מtabצע בהתאם להסתברות הגבוהה ביותר המתאימה לכל פיקסל.

אימון הרשת בוצע על בסיס מידע קטן יחסית של 600 תמונות דרך צבעוניות בגודל 480×360 , שנלקחו מבסיס המידע CamVid road scene dataset. סט האימון הכיל 367 תמונות וotto הבדיקה הכליל את התמונהות הנורטור. המטריה הייתה להזות בתמונהות אלה 11 חלקיות (דרך, בניין, מכונית, הולכי רגל וכדומה). כל תמונה עברה נרמול מקומי לערך הניגודיות של תמונה הקלט לפני הכניסה לרשת. האימון בוצע בשיטת ה-SGD, עם קצב למיניה קבוע שערכו 0.1 ומומנטום שערכו 0.9. האימון נמשך עד ששגיאות האימון התקטסהה. לפני כל epoch האימון ערבב וחולק ל-*mini-batch* של 12 תמונות. פונקציית המחרה הינה:

לעתים, נדרש לבצע איזון-מחלקות (class balancing). מונח זה מתייחס לכך בו קיימים שינוי גדול בין כמות הפיקסלים המשייכים לכל מחלקה, למשל כאשר קיימת מסוימת - סיכוי שברובה מחלקה בניינים / דרכים. במקרה זה, יבוצע משקל חדש לפונקציית השגיאה, באמצעות תהליכי "איזון התדריות החזיניות" (median frequency balancing). התהליכי משקל מחדש את המחלקות בפונקציית המחר, באופן ייחודי לחלקן של תדריות והופעות המחלקות בכל סט האימון, תוך חילוקה בתדריות הופעת המחלקה:

$$\alpha_c = \frac{\text{median freq}}{\text{freq}(c)}$$

משמעות זה משנה את היחסים בפונקציית המחיר כך שהתרומה של כל המחלקות לפונקציית המחיר תהיה שווה. לכן, הוא מעניק למחלקות הגדלות יותר משקל נמוך יותר ולחולקות הקטנות משקל גבוה יותר.

9.2.3 Atrous Convolutions (Dilated Convolutions)

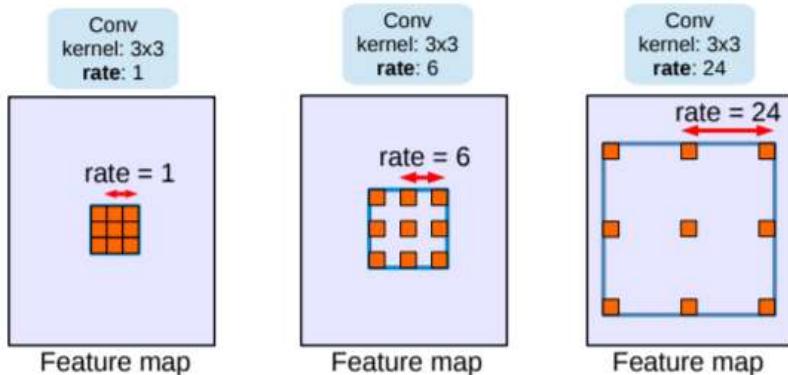
המושג Atrous מקורה בשפה הצרפתית – "à trous", שמשמעותו "עם חורים". כלומר, ניתן לתרגם את המונח Atrous convolution כ-"קונבולוציה מחוררת", ובמשמעותו מעט יותר מתאימה – קונבולוציה מרוחקת (או בשם אחר – Dilated convolution – קונבולוציה מרווחת).

בเทคนיקת קונבולוציה זו, יש שימוש בפרמטר נוסף בנוסף להקונבולוציה – dilation rate. פרמטר זה מסמן את המרווח בין כל איבר בגרעין הקונבולוציה (הרחבה על פרמטר זה – בפרק 5.1.2). גוסחת הקונבולוציה עבר המקירה החד מדי יחד עם פרמטר ההתרחבות r ניתנת לתיאור באופן הבא:

$$y[i] = \sum_{k=1}^K x[i + r \cdot k]w[k]$$

עבור $r = 1$ מקבלת הקונבולוציה הרגילה, ואשר ישנו גוסחת קונבולוציה מרוחקת בפקטור r . יתרונה של קונבולוציה נוץ בכך שהיא אוטומטית קרנל ובעור אותה כמהות חישובים, מרחיבים את ה- field of view (FoV) של הקונבולוציה.

ניתן לראות את הרחבת field of view באירור הבא:



איור 9.10: קונבולוציה מרוחקת ד-ממדית, עם קרנל בגודל 3×3 ופרמטר ההתרחבות $r = 1,6,24$. בהתאם לכל פרמטר מתקבל field-of-view שונה – בגודל שונה – $3 \times 3, 16 \times 16, 49 \times 49$.

9.2.4 Atrous Spatial Pyramidal Pooling

9.2.5 Deeplab V3+

נוקח רגע אחד אחריה בעת שנגידיר את שלוש הביעות המרכזיות של שימוש ברשותת קונבולוציה عمוקות (Deep convolutional neural networks :Semantic segmentation)

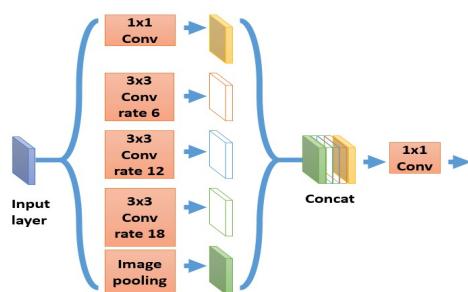
1. **מצאים ברזולוציית הפיצרים** – קיימות שתי סיבות מרכזיות לתופעה זאת. הסיבה הראשונה היא שאימון של Semantic segmentation מורכב ויקר בהרבה מאשר קלסיפיקציה לתמונה. אך, נעדיף להשתמש ב-transfer learning מאשר ברשות שאומנה זה מכבר ובועלת ידע קודם. יש לכך יתרונות וחסרונות, היתרון העיקרי הוא ההפחיתה המשמעותית בעליות. לעומת זאת, אחד החסרונות הוא שרוב הרשות אומנו על Imagenet dataset שמכיל תמונות ואובייקטים רבים, אך התאמונתו בו בעלות רזולוציה נמוכה, מה שעלול להוביל לרזולוציית פיצרים נמוכה.

הסבירה השנייה היא שכאשר אנו משתמשים ברשתות קונבולוציה عمוקות אנחנו מנוסים לקבל הינה לבני האובייקט שמסתתר לנו בתמונה. על ידי זיקוק האובייקט מתוך התמונה הגדולה נוצרת באופן אינרנטי רידה ביחסויציות הפיצרים.

2. **הימצאות של אובייקטים בגודלים שונים** – ניקח לדוגמא תמונה בה מופיעה מכונית גדולה מאוד ומוכנית קטנה. במרקם מסווג זה, הרשת תנסה לשיר את שני האובייקטים ולהלן לאוונה קטגוריה.
3. **הפחטה בדיקת המיקומי בכלל השונות של רשתות קונבולוציה عمוקות** - במקרים אחרות, רשתות קונבולוציה עושות עבודה מעוללה בזיהות דפוסים, אך יש פה סוג של trade off - הרשת תזהה את הדפוס הכללי אבל "תשכח" איפה בדיקת מופע האובייקט בתמונה.

כותבי המאמר של V3+-DeepLab תיארו שיטות להטבות עטיפות עם הביעות שתיארנו. הם השתמשו בשתי ארכיטקטורות מרכזיות, חיברו אותן ומספר שיפורים יצרו רשת חזקה עם תוצאות יפות על ה-datasets המרכזיים.

נדון כעת בשתי השיטות המרכזיות שהופיעו במאמר -



איור 1 (ASPP) Atrous Spatial Pyramid Pooling - 1

השיטה הראתה היא **קונבולוציה מרוחקת** (Atrous convolution) כפי שנכתב לעיל, מדובר במקרה בעל ממדים מוגבל. כל זה מאפשר לנו לצלם מידע קוטקסטואלי עשיר ביותר "ע" איגוד של פיצרים בגודלים שונים. כאשר אנו עושים שימוש במרוחקים שונים אנו מקבלים ראייה מרחבית משתנה ולא קבועה, איגוד התמונה לתמונהichert מהאפשר לרשף כוח אדיר של הينة. באמצעות זאת, הרשת מסוגלת ליצור קרשים בין אובייקטים שונים ואך בין אותו אובייקט שמופיע בגודלים שונים בתמונה.

הaintואיזיה שעומדת מאחורי ASPP היא שכאשר אנו עושים קונבולוציה רגילה אנחנו בוחרים להתריכז בחלק מאוד ספציפי (בגודל קבוע) בתמונה. באופן זה מקבל את הפיצרים מאותו החלק אבל ההקשר הסביבתי יהיה מוגבל. זאת לעומת שימוש ב- ASPP אשר מפנה הקשר סכמטי שונה וגדול יותר (בגודל הפרמטר של המרווח המשתנה). כמובן, כאשר מאגדים את התמונות לתמונה אחת אנחנו מחזקים את ההבנה הגלובלית שיש לרשת על התמונה. זאת לאור העובדה שכבר לא מודבר בהסתכלות על תמונה אחת אלא על הרבה זוויתות שונות שעזרת לרשת להבין קרשים וקשרים בצורה יותר ברורה. נאין באיר 2 שלפנינו -

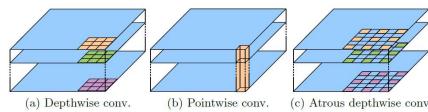


איור 2 – דוגמא לשימוש בשיטה

אם נרצה שהרשת "תסתכל" רק על הכליל המופיע ברכיב הערך שאותו מודובר במכונית. אך ברגע שהרשת "תסתכל" את המים מסביב ואת התמונה הרחבה יותר -- היא תקבל הבנה عمוקה יותר וכן תבין שמדובר בסירה ולא במכונית. דבר דומה יתרחש כאשר יהיה מופיע בתמונה של אותו אובייקט בגודלים שונים: האיגוד של התמונה עם המוחוכים השונים עוזר לשרת קשר בין אותו אובייקט.

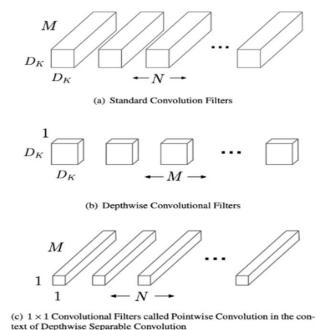
מבחן טכני, הפעולות שאנו מבצעים ב- ASPP על התמונה מורכבות מ- 4 קונולוציות מרוחקות. בכל אחת מהן אנו מקבלים תמונה באותו הגודל, אבל בעלת ראייה מרוחבית שונה. שימושים שונים לאמור השתמשו בפונקציית אקטיביזציה (ReLU).

בהמשך לביצוע הקונולוציות, נבצע פעולה pooling על התמונה. מימושים שונים לאמור השתמשו ב- Adaptive average pooling על מנת שנוכל לחזור לגודל התמונה ולבצע שרשור בין כל התמונות.



Atrous Separable Convolution - 3

בהתיבט של עליות חישוב, כתבי המאמר משתמשים בטכנית **Atrous Separable Convolution** שמטرتה להוריד בזרה ניכרת את מספר הפרמטרים והчисובים שהרשת צריכה לבצע.



Separable convolution - 4

- Depthwise convolution – מכך בתוכה שתי טכניקות עיקריות – **Separable convolution** ו-**Pointwise convolution**.

סביר את depthwise convolution: אנו מבצעים את הפעולות על מרחב מידע התמונה (ציר ה x ו-y) ו-
ב-convolution pointwise מבצעים את הפעולות על מרחב העורף.

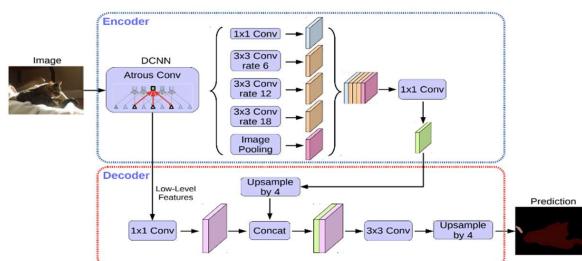
מבחן מתמטית, נניח שה- $F \in D_F \times D_F \times N$ input feature maps, ושה- $G \in D_K \times D_F \times N$ convolution kernel. כדי להיות בוגד $D_K \times D_F \times N$ 我们必须使用 convolution kernel regularization,我们需要在计算过程中引入一个惩罚项，以限制 kernel 的大小。为了使计算更高效，我们可以在卷积核上应用深度分离卷积（depthwise convolution），这样可以将卷积操作分解为两个部分：深度分离卷积（depthwise convolution）和点积卷积（pointwise convolution）。深度分离卷积将输入特征图的通道数减少到 D_K ，而点积卷积将 D_K 个通道的特征图进行点积操作，从而得到输出特征图。

אבל אם נבחר להשתמש ב- $conv$, ראות נבצע $conv$ על מרחב המידע $D_F \times D_F \times N$. כלומר, $K \in D_K \times D_F \times N$. לכן, במאובן זה אנו מבצעים $D_F \times D_F \times N$ פעולות $conv$ על מרחב המידע $D_F \times D_F \times N$. לאחר מכן נבצע $pointwise conv$ על מרחב המידע $D_F \times D_F \times N$. אם פילטר הקונבולוציה במקורה זה יהיה בגודל $1 \times 1 \times M$ וכאן עלות החישוב תהיה $N \times M$. אם נחבר את שתיים עלות החישוב תהיה:

$D_F \times D_F \times D_K \times D_K \times M + D_F \times D_F \times M \times N$, ניתן לראות שבניגוד לكونבולוציה הרגילה, במצב זה נוצר חיסכון משמעותי בעלות החישוב ובמספר הפרמטרים.

כasher נשתמש בكونבולוציה מרווחת נקל על חישוב תחיה: **Atrous separable convolution**

ארQUITקטורת מקודד מפענה (encoder decoder)



אייר 5 - ?

نبיט על ארQUITקטורת המקודד ונראה שכותבי המאמר למעשה העתקו את המקודד שבו השתמשו בראשת DeepLab V3+: makodod mocholak le-3 chlikim:

1. רשת backbone שמרתת החלט פיצרים מהתמונה - במאמר משתמשים בורסיה של Xception (נראה עלייה בהמשך). בסוף לכך רשות backbone מחזירה גם low level feature low level feature低层特征.(entry flow) לאחר ה-block הראשון של ה-
2. לאחר מכן אנו מבצעים ASPP על התמונה.
3. לבסוף ביצוע קונגולוציה 1X1 על מנת לעבור למספר העורכים הרצוי.

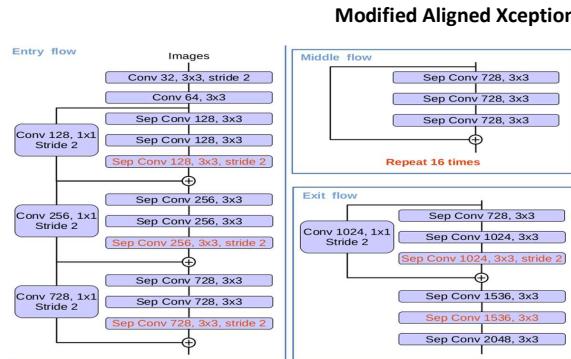
לעומת זאת המפענה עבר שינוי דרמטי מכיוון שכותבי המאמר רואו שהשחזור של התמונה לגודל המקורי לוגה בחסר ופוגע ביכולת הדיק של הרשת, שלוש הטכניקות שבהן הם השתמשו:

1. שימוש ב- low level feature לשיזוע השחזור.

2. שימוש ב- upsample בהדרגה.
3. הפעלת קונволוציה 3×3 על מנת לקבל אובייקטים חדשים וברורים יותר.

מבחןת טכנית הסכמה של המענה מtbody כ:

- מתקבלת תמונה מהמקודד ועליה ניתן לבצע upsample פי 4 (לדוגמא עם התמונה היא בגודל 32×32 היא עכשו תהיה בגודל 128×128).
- לאחר מכן, יש לבצע קונволוציה 1×1 על הקלט שהתקבל מה- low level feature על מנת להקטין את גודל מימד העורק.
- לאחר מכן, יש לבצע שרשור של התמונה עם ה- low level feature.
- יש לבצע קונволוציה 3×3 כדי חזק את האובייקטים שבתמונה בנוסף, מספר העורקים שבתמונה ישתנה למספר הקטגוריות שיש לנו ב- dataset.
- לבסוף יש לבצע upsample נוספת בגודל פי 4 על מנת לחזור לגודל המקורי של התמונה.



איור 6 Modified Aligned Xception

כותבי המאמר בחרו לשימוש ברשת זו כ backbone בغالל ש- Xception ידועה ביצועים טובים ומהירים במשימת הקלסיפיקציה ועם עליונות מסוימת. כמו כן, Xception מאופיין ביכולתו לזרום מושג של נזק במשימת ה- object detection.

- במקומם max pooling או מבצעים convolution atrous stride 2 עם depthwise convolution.
- לאחר כל ביצוע של convolution 3×3 אנו נבצע נרמול וקטיביזציה.

ביצועים

מצלחים לייצר state of the art DeepLab V3+ ב- MIoU 89 עם PASCAL VOC 2012 Test Set חדש גם ב- Cityscapes dataset ב- MIoU 82.1 עם

9.3 Face Recognition and Pose Estimation

9.3.1 Face Recognition

אחד מהישומים החשובים בראייה ממוחשבת הינו זיהוי פנים, כאשר ניתן לחלק משימה זו לשולש שלבים:

1. Detection – מציאת הפרצופים בתמונה.
2. Embedding – מיפוי כל פרצוף למחרב חדש, בו המאפיינים שאינם קשורים לתיאור הפנים (למשל: זווית, מיקום, תארה וכדו) אינם משפיעים על הייצוג.
3. Searching – חיפוש במאגר של תמונות למציאת תמונה פנים הקרובה לתמונה הפנים שוחלצתה מהותmana המקורית.

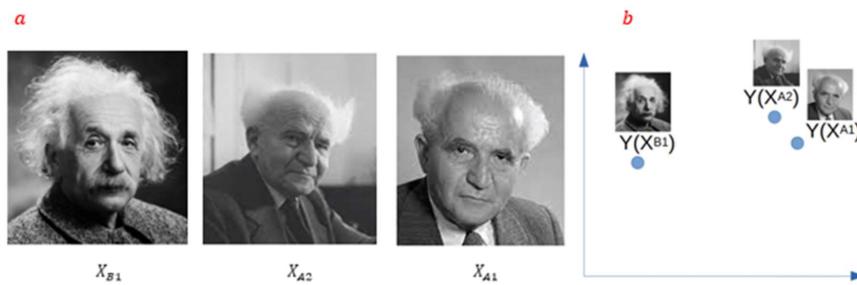
גישה פשוטה, כמו למשל ביטת מושג המכיל סדר ייצואו כמספר הפנים אוזם רואים ליהו, הינה בעיית משתן סיבוב עיקריות: ראשית יש צורך באלאי דוגמאות לכל אדם (שלא ניתן בהכרח להציג). כמו כן, נוצרר למד את המערכת מחדש בכל פעם שורצים להסיף מישחו חדש. כדי להתגבר על בעיות אלו מבצעים "למידת מטריקה" (metric learning) בה מזקקים מאפיינים של פנים ויזרים וקטור יחסית קצר, למשל באורך 128, המכיל את האלמנטים הרוכשיים בתמונה הפנים. עת נפרט את שלושת השלבים:

1. מציאת פנים.
- כדי למציאו פרצופים בתמונה ניתן להשתמש ברשותות המבצעות detection, detection, כפי שתואר בפרק 9.1. שיטה מקובלת למשמה זו הינה סולו, המבוססת על חילוק התמונה למשבצות, כאשר עבר כל משבצת בחונים האם יש בה אובייקט מסוים, מהו אותו אובייקט, ומזה ה-*bounding box* שלו.

2. תיאור פנים.

כאמור, המשימה בתיאור פנים נעשית באמצעות metric learning, כאשר הרעיון הוא לזרק פנים ללקטור שאנו מושפע ממאפיינים שלא שייכים באופן מהותי לפנים הפסיכיות האלה, כגון זווית צילום, רמת אורה וכו'. כדי לשנות זאת יש לבנות רשת המקבלת פנים של בנאדם ומחדרה וקטור, כאשר הדרישה היא שעבור שתי תמונות של אותו אדם יתקבלו וקטוריים מאד דומים, ועבור פרצופים של אנשים שונים יתקבלו וקטוריים שונים. למעשה, פונקציית ה-*loss*-*batch* בכל פעם *minibatch*, ותעניש בהתאם לקרבה בין וקטוריים של אנשים שונים ורחוק בין וקטוריים של אותו אדם.

עתה נניח שיש לנו קלט X , המכיל אוסף פרצופים. כל איש יסמן באות אחרת – A, B, C , ותמונה שונות של אותו אדם יסמננו על ידי אותן ומספר, כך למשל X_{A1} זהה התמונה הראשונה של אדם A בסט הקלט X , וכמוון X_{A2}, X_{B1} הןשתי תמונות של אותו אדם. באופן גראטי, בוד-המודד ניתן לתאר זאת כך (בפועל הווקטורים המייצגים פנים יהיו במד גובה יותר):



איור 9.10 (a) דוגמאות מסט הפרצופים X . (b) אייר נרצה שהדדרטה 'מושפה' לממד חדש \mathcal{Y} .
 כאמור, נרצה לבנות פונקציית loss שמעודדת קירבה בין X_{A1} - X_{A2} , וריחוק בין X_{A1} - X_{B1} . פונקציית loss מורכבת משני איברים, המודדים מרחק אוקלידי בין וקטורים שונים:

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

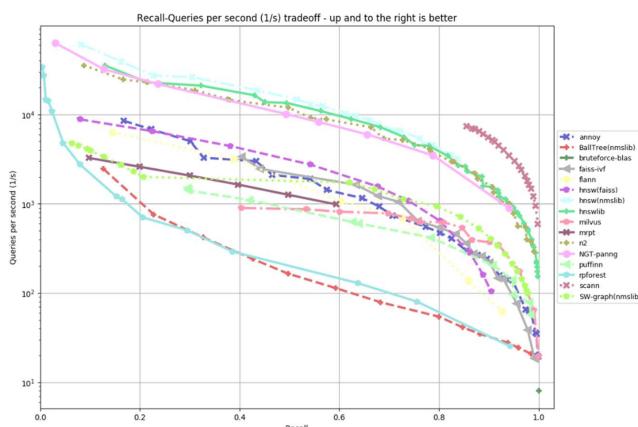
כאשר האיבר הראשון ינסה להביא למינימום וקטוריים של אותו אדם, והאיבר השני ינסה להביא למינימום וקטוריים של פרצופים שאינם שייכים לאותו אדם. כיוון שנרצה להימנע מתקבלות ערכיהם שליליים, נוסיף פונקציית מקסימום. בנוסף, ניתן להרוחק תוצאות של פרצופים שונים על ידי הוספת קבוע k , כך שהפרש בין המרחק של פרצופים של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

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

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

3. מציאת האדם

בשלב הקודם, בו התבצע האימון, יצרנו למשה מאגר של פרצופים במרחב חדש. כתע' כשייגע פרצוף חדש, כל שנתרן זה למפות זאת אותו למרחב החדש, ולהפוך במרחב זה את הווקטור הקיים ביוטר לווקטור המציג את הפנים החדש. בכך ניתן לעשות זאת באמצעות קלאלסיות של machine learning (כפי שהסביר בחלק 2.1.3). שיטות אלו יכולות להיות איטיות עבור מאגרים המכילים מיליון וקטורים, וישן שיטות חיפוש מהירות יותר (ובדרך כלל המהירות גבוהה על חשבון הדיוק). בעזרת השיטה המובילה כרגע (SCANN) ניתן להגיע לכמה מאות חיפושים שלמים בשניה (ההיפוש ב-100 ממדים מדורג לשולחן ב-10000 דוגמות).



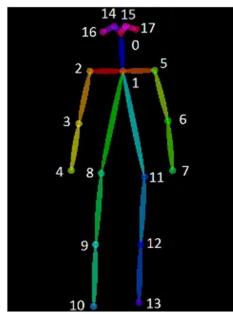
איור 9.11 השוואת ביצועים של שיטות חיפוש שונות. עבור פרצוף נתון, מוחפשים עבורו וקטוריים בממד החדש המכיל 'יצוג וקטורי' של הפרצופים הידועים. בכל שיטה יש טריד-אוף בין מהירות החיפוש לבין הדיוק.
מלבד זיהויoso ייזוג פנים, יש גם שיטות של מציאת אלמנטים של פנים הכלולות אף, עיביים וכו'. אחת השיטות המקובלות משתמשת בשעריך הצורה של פנים אונשיוט, וניסיון למצואו את איברי הפנים לפי הצורה הסטנדרטית. בשיטה זו ראשית מבצעים יישור של הפנים והתאמאה לסקלה אונשיוט (על פי מרחק בין האיברים השונים בפנים), ולאחר מכן מטילים 68 נקודות עניין מרכזיות על התמונה המיישרת, מטור ניסיון להתאים בין הצורה הידועה לבין התמונה המבוקשת.



איור 9.12 זהוי אזוריים בפנים של אדם על ידי התאמת פנים לסקללה אונושית והשואה למבנה של פנים המכיל 68 נקודות מרכזיות.

9.3.2 Pose Estimation

ישום פופולרי נוספת של אלגוריתמים השיכים לראייה מוחשבת הינו קביעת תנוחה של אדם – האם הוא עומד או ישב, מה התנוחה שלו, באיזה זווית האיברים נמצאים וכו'. ניתן להשתמש בניתוח התנוחה עבור מגוון תחומיים – ספורט, פיזיותרפיה, משחקים שונים ועוד. לרוב, תנוחה מייצגת על ידי המיקומים של חלק גוף עיקריים כגון ראש, כתפיים, מפרקים וכו'. ישנו כמה סטנדרטים מקובלים, למשל COCO, posenet ועוד. ב-COCO התנוחה מייצגת בעדרת מערך של 17 נקודות (בדו-מימד):



איור 9.13 מיפוי תנוחה ל-17 נקודות מרכזיות.

שאר הפורמטים דומים; מוסיפים עוד מידע (למשל מיקום הפה), משתמשים בתלת ממד במקומ בדו ממד, משתמשים בסידור אחר וכו'. כאשר רוצים לאוסף נתונים על מנח גוף שונים, ניתן לשים חישונים על אותן נקודות מרכזיות וכן לקבל מידע על מנת הגוף לאורך זמן.

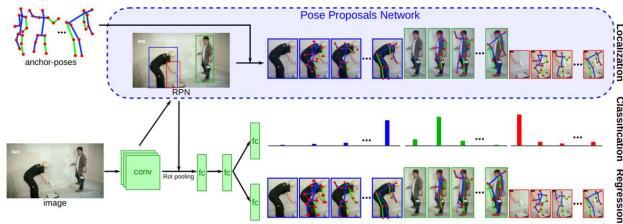
רשות לצורך קביעת תנוחה יכולה לטפל במרקחה כלל, בו יש מספר אנשים בתמונה, או במקרה הפרטי של גוף יחיד. המקרה השני כמבודן יותר פשוט, מכיוון שהוא מושם פלט ייחד, ואנו ניתן לחזות באמצעות רגסיה (למשל, 34 מספרים שמתארים את המיקומים של 17 הנקודות בפורמט COCO בדו-ממד). במקרה זה ניתן לעשות שימוש סטנדרטית לחולוטן של בעיית רגסיה בעלת 34 ציאות.

ישן מספר גישות כיצד להכליל את הרשות כך שתוכל לטפל גם במקרה הכללי בו יש יותר מגוף אחד בתמונה. באופן נאיבי ניתן לבצע תהליכי מיקדים של מציאת כל האנשים בתמונה, ואז להפעיל על כל אחד מהם בנפרד את הרשות שמבצעת רגסיה, כדי שתוארך לעיל. שיטה נוספת נוספת פועלת בכיוון הפוך – ראיית כל הרשות מוצאת את כל האיברים בתמונה נתונה, ולאחר מכן משיכת אותם לאנשים שונים. השיטה השנייה נקראת "מלמתה למעלה" (top-down), כיון שהיא שוקודם כל האיברים וஅחר מכך משיכת אותם. גישה זו עיליה למקרה בו יש הרבה חfine בין האנשים בתמונה, כיון שהיא צריכה לתקן תהליכי מיקדים של הfrared האנושיים. השיטה הראשונה, הקרויה "מלמתה למטה" (down-top), תהיה פשוטה יותר עבור מקרים בהם אין חfine בין האנשים בתמונה וכן אינן מחייבת מושג אחד מהם נמצא באזורי שונה בתמונה, כיון שבמרקחה זאת אין צורך לשיר איברים לאנשים.

רשות פופולרית לקביעת תנוחה נקראת pose estimation, המורכבת למעשה משתי תת-ရשותות ופעלת בשיטת top-bottom. הרשות שאחראית על שייר חלק גוף לאדם מסוים, נקראת PAF, part affinity fields, והרעיון שלו הוא ליצג כל איבר כshedde וקטורי. ביצוג זה הוקטוריהם השונים מצבעים לכיוון איבר הגוף הבא בתווך (למשלذرוע מצבעה לצד), וככה ניתן לשיר איברים שונים אחד לשני, ואת כל גוף מסוים.

רשת פופולרית אחרת, הפעלת בגישה通道-top, נקראת LCR-NET, והיא מבוססת על רעיון של 'מיקום-סיווג-רגסיה' (Localization-Classification-Regression). בשלב הראשון יש תתרשת המיצרת עוגנים עברו אنسים, ככלומר, אזוריים בהם הרשת חשבת שמאן בן אדם, ולאחר מכן הרשת משערת את התנוחות שלהם. בשלב השני מתבצע clustering לכל העוגנים, וכך כל עוגן מקבל ציון המציג את טיב השערוך של העוגן והתנוחה של האדם הנמצא בתוכו. בשלב השלישי מילש את העוגנים ומשקלה את השערוך הסופי בעדרת מיצוע של הרבה העוגנים.

שלושת השלבים משתמשים ברשת קובולוציה משותפת, כמפורט באירא.



.אייר 9.14 Localization-Classification-Regression 9.14

9.4 Few-Shot Learning

יכולת הצלחתם של אלגוריתמי למידה עמוקה נשענת על כמות ואיכות הדата לאימון. עבור משימות סיווג תמונות (Image Classification), נדרש כל קטגורית סיווג תריה כמות גודלה של תמנונת (שם הבדלי רקיעם, בהירות, זווית וכו'), ובנוסף יש צורך בכמות דומה של דוגמאות בכל קטגוריות הסיווג. חומר איזון בין כמות התמונות בקטגוריות השונות משיער על יכולת הלמידה של האלגוריתם את הקטגוריות השונות ועל כן עלול ליצור הטיה בתוצאות הסיווג לטובת הקטגוריות להן יש יותר דוגמאות. פררך זה עוסק בשיטות כיצד ניתן להתמודד עם מצבים בהם הדטה לאינו מואן.

9.4.1 The Problem

התמונה של למידה ממיעוט דוגמאות (Few-Shot Learning) נוצר על מנת להתמודד עם מצב של חומר איזון קיצוני בין כמות הדוגמאות של כל קטgorיה לאימון הרשת. באופן פרטני, קיימות קטגוריות הנקראות בסיס (base classes), עבורן יש כמות גדולה של דוגמאות, ובנוסף ישן קטגוריות חדשות (novel classes), עבורן יש כמות קטנה מאוד של דוגמאות. בכך לאגדיר את היחס, משתמשים בשני פרמטרים: פרמטר A המציג את מספר ה-shots, כלומר מספר דוגמאות קיימות בסיס האימון מכל קטגוריה דחונית, ופרמטר B ש מייצג את מספר הקטגוריות החדשניות הקיימות בסך הכל. כל בעיה ווגדרת על ידי "one-shot k-way learning", ולמשמעותו מתייחס ב- k -shot 5-way learning, מצלב אחד מהן יש רק דוגמא אחת לאימון הרשת. ככל, בקטגוריות הבסיס תהיה כמות גדולה של דוגמאות. למשל בסיס התמונות האופני לבניית אל, mini-ImageNet, יש 600 דוגמאות לכל קטgorיה בסיס ורובה 5-1 דוגמאות עבור קטגוריות החדשניות.

האתגר בלמידה ממיעוט דוגמאות נבע מה הצורך להכין לרשת כמות ידע קתינה נוספת על הידע הנרחבות הקיימ, תוך הימנע מ-overfitting כתוצאה מכמות הפרמטרים הגדולה של הרשת לעומת הכמות המועטה של הדטה. לכן, גישה איבית כמו אימון מחדש של רשת על מעת דוגמאות נוספת לצורך התחזוקה.

יש לציין כי בכל בעיות הלמידה ממיעוט דוגמאות, אף אחד חסר DATA עבור האימון, אך השאיפה היא להצליח באופן זהה בחיהו כל הקטגוריות בשלב המבחן, בו לא יהיה חומר איזון. לכן בעיות אלו רלוונטיות לשימושים רבים כמו: זיהוי חיות נדירות באופן זהה לחיות יונת נפוצות, מערכת זיהוי טילים ש.'

צריכה להתמודד גם עם אינדים יותר (יתן לחשב למשל על פצצת אטום), מערכות זיהוי פנים ש.'

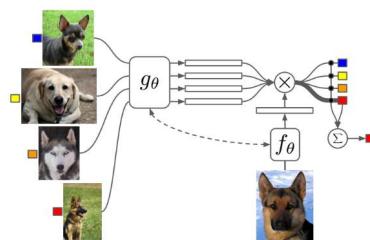
צריכה לעבוד טוב עבור כל אדם ללא תלות בדתא שהיא קיימת באימון הרשת.

פרק זה נתאר את שלוש הגישות העיקריות לפתרון בעיות למידה ממיעוט דוגמאות. עבור כל גישה נציג את האלגוריתמים המשמעוניים ביותר שנקטו בגישה זו. לאחרונה, מפותחים יותר ויתר אלגוריתמי למידה ממיעוט דוגמאות שימושיים יחד ריעונות השאובים ממספר גישות יחד אך נשיינים על האלגוריתמים המשמעוניים מהעבר. לבסוף, נציג את התהום של Zero-Shot Learning, וכך יכולת למידה של קטגוריה חדשה כאשר לא קיימת אף דוגמא שלא לאימון.

9.4.2 Metric Learning

שיטות להתמודדות עם למידה ממיעוט דוגמאות הנקוטות בגישה למדית מטrique, שואפותלייצג את הדוגמאות כוקטורים של מאפיינים במרחב רב-ממדי, כך שניתן יהיה למצואו בклות את השיר הקטגוריה של דוגמא חדשה, גם אם היא תהיה מקטגוריה חדשה. שיטות אלו מבוססות על עיקון הגדלת המרחק בין יציגים קטררים של דוגמאות מקטגוריות שונות (inter-class dissimilarity), ובבבשמייה על מרחק קטן בין הייצוג הווקטורי של דוגמאות מאותו הקטגוריה (intra-class similarity).

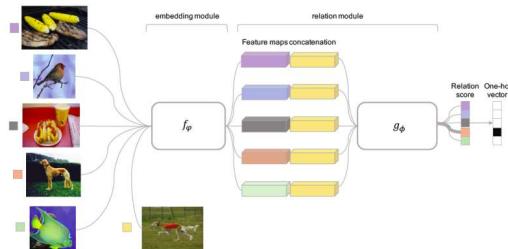
התקרחות משמעותית של שיטות אלו הוצאה במאמר Matching Networks for One Shot Learning בשנת 2016. שיטה זו משתמשת בזיכרון שהגישה אליו נועשית באמצעות מגנון, Attention, על מנת לחשב את ההסתברות של דוגמא להיות שייכת לכל קטגוריה, בדומה לשיטות השכן הקרוב (Nearest Neighbors).



.איור 9.15 אילוסטרציה של שיטת Matching Networks

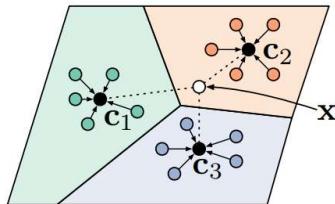
החדש המשמעותי בשיטת האימון Matching Networks נעה בשיטת האימון המבוצע באפיוזות (Episodes). בשיטה זו הAIMON מכיל כמהות של מושימות, כאשר כל מושימה היא למעשה מוגם של דאטה שבעקבות מושימות שנערכו יחדיו ואחרות שנערכו נפרד. על ידי דגימות רבות ויצירת שיטות מלאכותיות כאלה, בהן בכל פעם נלקחות קטגוריות אחרות ליציג את החדשניות, מתבצע אימון המתאים לבעה של מיעוט דוגמאות. שיטת אימון באפיוזות הפכה לפופולרית במיוחד במהלך מילדייה מיעוט דוגמאות, גם בגישה המטrique וגם בגישה שרואה בהמשך.

שיטות רבות מותבססות על הרעיון של מאמר זה. למשל שיטת Relation Network משרשת וקטורי מאפיינים של דוגמת מבחן בין כל דוגמא של קטגוריות האימון. אלו נכונים למודל המשער ממד דמיון בעדרתו ניתן לסוג את דוגמת המבחן.



.Learning to Compare: Relation Network for Few-Shot Learning 9.16

שיטה נוספת נוספת.. Prototypical Networks. בגישה זו כל קבוצה של דוגמאות של קטגוריה מסוימת במרחב וקטורי המאפיינים מקבלת נקודת אבטיפוס אופיינית המוחשבת על ידי הממוצע של הדוגמאות בקטגוריה זו. בכך מוחשיים מסווג יינארה המפheid בין הקטגוריות. בעת המבחן מסווג דוגמא חדשה על סמך מרחק אוקלידי מנקודות האבטיפוס.



.איור 9.17 Prototypical Networks for Few-Shot Learning

בטבלה הבאה ניתן לראות השוואת ביצועים של שיטות למידת המטריקה שהוזכרו על הקטגוריות החדשניות. יש להזכיר כי כל השיטות מגיעות לאחוזי דיוק נמוכים משמעותית מאשר הדיוק המדויקים במרקם של איזון בין כמות הדוגמאות בקטגוריות השונות (לרוב מעל 90% דיוק).

Method	5-way 1-Shot	5-way 5-Shot
Matching Networks	46.6%	60.0%
Prototypical Networks	49.42%	68.20%
Learning To Compare	50.44%	65.32%

.איור 9.18 השוואת ביצוע דיוק של שיטות למידת מטריקה על קטגוריות חדשניות עבור mini-ImageNet

9.4.3 Meta-Learning (Learning-to-Learn)

גישה שנייה להתרמודדות עם מיעוט דוגמאות וחוסר איזון בין הקטגוריות נקראת מטא-למידה (או: לומוד איך ללמידה). באופן כללי בלמידה מכונה, כאשר מדובר על מטא-למידה, מתקווים לרשף שלומדת על סמן התוצאות של רשות אחרת. בלמידה ממיעוט דוגמאות הרעיון הוא שהרשות תלמד בעצמה איך להתרמודד עם מיעוט הדאטה על ידי עדכון הפרמטרים שלא לאופטימיזציה של בעיה של סיווג ממיעוט דוגמאות. לשם כך משתמשים באפשרות של מושמות לmeta-למידה ממיעוט דוגמאות.

שיטה החשובה בגישה זו היא MAML (Model-Agnostic Meta-Learning). בשיטה זו, שאינה מיועדת ספרטיפית לשינוי תמנוגות ממיעוט דוגמאות, בעדרת מספר צעדים מעטים בכיוון הגרדיאנס ניתן למדו את הרשת התאמת מויילה (fast adaptation) למשימה חדשה. לצורך, כל משימה באיזומורפיזם שבסה קטגוריות מסוימות נבחורת רנדומלית לדוגמאות הדרישות החדשניות. בכל משימה צדו נלמדים פרמטרים של המודול האגנוטיסטי כך שעדכונם בכיוון הגרדיאנט יוביל להתאמה למשימה החדשה. הכותבים מציינים שמנקודות מבט של מערכות דינמיות, ניתן להתבונן על תהליכי הלמידה שלהם ככזה שמקסם את רגישות פונקציית המחיר של מושמות חדשות ביחס לפרמטרים. כאשר הריגשות גבוהה, שניי פרמטרים קטנים יכולם להוביל לשיפור מושמות מחיר של המשימה. מתמטית, פרמטרי המודל, המיוצגים על ידי θ , מושתנים עבור כל משימה i להיות θ'_i , כאשר:

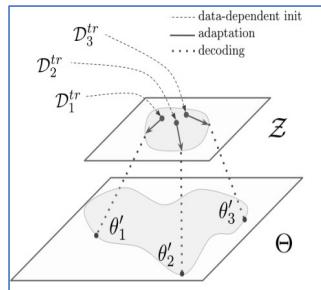
$$\theta'_i = \theta - \alpha \nabla_{\theta} L_{T_i}(f_{\theta})$$

עבור פונקציית מחיר L והיפרפרמטר α . כאשר מבצעים מטא-למידה לעדכון הפרמטרים, מחשבים למשעה SGD:

$$\theta \leftarrow \theta - \beta \nabla_{\theta} \sum_{T_i \sim p(T)} (L_{T_i}(f_{\theta}))$$

כאשר המשימותxDוגמאות מתווך (T) a - β הוא גודל הצעד של המטא-למידה.

שיטה מעניינת נוספת בשם Latent Embedding Classification (LEO) מימושת את הגישה של מטא-למידה במרחב יזוג לטני בעל ממדים נוספים, ולאחר מכן עבר בזרה למרחב המאפיינים הרב ממד.



. איור 9.19 שיטת (LEO)

השימוש במרקח מאפיינים בעלי מדדים נומיים המשמשים את המאפיינים החשובים לייצוג הקטגוריות, שיפור באופן ניכר את תוצאות הסיווג, כפי שנitin לראות בטבלה הבאה:

Method	5-way 1-Shot	5-way 5-Shot
MAML	48.7%	63.11%
LEO	61.76%	77.59%

. איור 9.20 השוואת ביצוע דיק של שיטות מטא-למידה על קטגוריות חדשות עבור mini-ImageNet עבור

9.4.4 Data Augmentation

גישה שונה להתחמיזות עם מיעוט דוגמאות נוקטת ביצירת דוגמאות כדי להימנע מהטיה. שיטות אוגנרטיביה למשה יוצרות DATA חדש על סמך הدادה המקורי. השיטות הפשטות יותר מיצירות מהתמונה המקוריות תמיון ראי, שינוי תאוורה וקונטרסט, שינוי סקללה, שינוי דזיות, ואף הוספה ראש רנדומלי. כל אלו הרואו שימוש ביכולות הרשות ללמידה קטגוריות שהוא במצב של חוסר איזון. דרך נוספת היא שימוש ברשותות גנרטיביות (GANs) על מנת ליצור דוגמאות רלוונטיות, למשל דוגמאות של אותו האובייקט מזוויות שונות. שיטה מעניינת של אוגננטציות היא MixCut, בה פאיים של תמיונות נחכמים ומודבקים בתמיונות האימון גם התייגים מעורבים בהתאם. שיטה זו הגיעו לביציעים מרשימים בסיווג תמיונות וגם ביזוי אובייקטים, ככל הנראה בגלל שהיא מאפשרת למודל להיות גנרי יותר בהתחייבות לחילוקים שונים מהתמונה המשפיעים על הסיווג לקטגוריה.

9. References

Detection:

<https://arxiv.org/pdf/1406.4729.pdf>

Segmentation:

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

SegNet:

<https://arxiv.org/pdf/1511.00561.pdf>

<https://mi.eng.cam.ac.uk/projects/segnet/#demo>

<https://arxiv.org/pdf/1409.1556.pdf>

<https://arxiv.org/pdf/1502.01852.pdf>

Face recognition:

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

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

10. Natural Language Processing

ההוטומטיות בתיבת החיפוש-*Google*-*Grammarly*, מערכתי לתיקון תחבירו לטעט בענגלאיט.

10.1 Language Models and Word Representations

בפרקם הקוראים דן בשנים המונחים הבסיסיים ביוטר בתחום ה-NLP – מודל שפה ויצוגי מילים: ראה כיצד מייצרים מודל שפה מדatta טקסטואלי ("Corpus", וכך מיצגים את הטקסט כך שהיא מאפשרת לאמן בעדרתו מודל, כדי להקנות למוניה יכולת הבנה של דата טקסטואלי יש לבנות מודל הסתברותי הקובע את ההסתגלות של המילים בשפה. המודל מנסה לemat את הסיכוי להופעה של סדרות שונות של מילים, ובאופן יותר כללי הוא קובע מה הסתברות של כל רצף אפשרי להופיע. מודל זה יכול לבנות בעדרת אימון על גבי דата טקסטואלי, והוא נקרא "מודל שפה" (Language Model). לפני ביצוע האימון, יש לבצע מיפוי של הטקסט ליצוג מושגים (Word Representation), המאפשר למוניה ללקחת את הדאטה הקיימ, לעבד אותה ולנסות לבנות בעדרתו את מודל השפה.

הארהית נגידר מהו מודל סטטיסטי המגדיר התפלגות מעל המורכב מכל הסדרות האפשריות של מילים (למשל משפטיים, פסקאות ועודומה). מודל זה מקבל כקלט סדרה של מילים ותקמידו הוא לחזות מה היא המילה הבאה שתנבי את ההסתברות המרבית לרצף ביחיד עם המילה הנוספת.

icut נתאר בזורה מתמטית מהו מודל שפה. נניח ונתן משפט עם n מילים: $[w_1, \dots, w_n]$, אז ההסתברות לקבל את המשפט הזה הינה:

$$P(w_1, \dots, w_n) = \prod_{i=1}^n P(w_i | w_1, \dots, w_{i-1})$$

ניכון למשל את המשפט הבא:

Take a big corpus

ההסתברות של משפט זה ניתנת לחישוב אופן הבא:

$$P(Take, a, big, corpus) = P(Take) \cdot P(a|Take) \cdot P(big|Take, a) \cdot P(corpus|Take, a, big)$$

באופן הפשטי ביותר נוכל לשער את ההסתברויות האל באופן הבא: ניקח *kruskos* מספק גדול ו usher בעל N מילימטרים שונים, כמו למשל לערכיו ויקיפדיה, ופושט ונספור את כל המילימוט והצירופים שונים. בזאת, ההסתברות של כל מיליה תהיה אחות הפעמים שהיא מושפעה בטקסט, וההסתברות המותנית תחשב באופן דומה – על ידי מנית מספר המופיעים של צירוף כלשהו וחלוקה במספר הפעמים שהמילה עצמה מושפעה. באופן פורמלי נוכל לרשותם זאת כך:

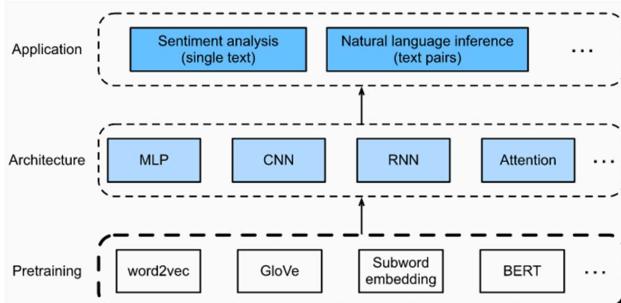
$$P(W_i) = \frac{\#W_i}{N}, P(W_i|W_j) = \frac{\#W_i, W_j}{\#W_i}$$

הבדלים העיקריים בין אמצעי תקשורת מודולריים וטלפון-fix נובעים מכך שטלפון-fix משלב מנגנון של ביצוע מושגים.

כאמור, כדי שוכן לבנות מודול שפה או לפחות כל מודול אחר, נצטרך קודם כל לyield את הטקסט בזורה לשלשא.

שראינו מרכיבת מצירוף של האותיות e, k, a, T. כפי שיפורט בהמשך, ככל גם לדבר על ייצוג למילים עצמן כך שכל יחידה אוטומטית תהיה מילה ויצירוף של היחידות האלה יריכיבו ייצוג של משפט.

באופן סכמטי, ניתן לתאר את מושגית עיבוד השפה מקצתו כך באפנ' הבא:



איור 11.1 תהילך פיתוח אלגוריתם של מודל שפה: א. מייצגים את הטקסט בצורה כלשהיא (ניתן כדוגמה ייצוג קי"ם שנבנה על בסיס דאטאטס אחר). ב. מאמנים מודל שקבלי-צטוקום את טקסט אותו ייצנו בדרך כלליהי בשלב הראשוני, והמודל מוציא utput קי"ם מסוים. למודל זה יכולות להיות ארכיטקטורות שונות. ג. באמצעות המודל המאומן ניתן לבצע שימושים קזה' שונים.

בהמשך פרק זה נתמקד בשכבה התחומרה של התרשים: נתאר מספר שיטות מרכזיות לייצוג טקסטים, ונראה כיצד ניתן לאמן מודלי שפה שונים היכולים לבצע כל מיני שימושים.

10.1.1 Basic Language Models

מודל השפה הראשון אותו נציג הינו **N-Grams**, שהוא מודל סטטיסטי המניח שהסתברות למילה הבאה תלויות אך ורק ב- $n-1$ מילים שקדמו לה בסדרה. הנחה זאת נקראת "הנחה מרקוב" (Markov assumption), ובאופן כללי יותר, מודלי מרקוב (או שרשרת מרקוב במקורה הדיסקרטני) מסדר n הם מודלי הסתברות המניחים شيئا'ן לחזות הסתברות של אירוע עתידי n בהתאם על האירועים שהתרחשו ב- $n-1$ נקודות הזמן שקדמו למאירוע n מוביל להתחשב באופן ערך רוחקים מדי.

מודל **n-Gram** הפשטוט ביותר נקרא **n-gram**. במודל זה אנחנו חוזים את המילה הבאה לפי התדריות של המילה עצמה **co-words** מוביל להתחשב במקרה שקדמו לה. מכיוון שהיחס בין המילים הבאה **חייב להיות תלוי במילים שקדמו לה**, כמו כן יהיו מילים כמו **the shmoofiyut** באופן תדייר בטקסט שאין בהכרח משפיעות על ההקשר. לכן, נסתכל על מודל קצת יותר מורכב הנקרא **n-gram**, המתיחס למילה الأخيرة הקודמת למילה הנחוצה. במודל **n-gram** אנחנו מניחים את המשווואה הבאה:

$$P(w_n | w_{n-1}, w_{n-2}, \dots, w_1) = P(w_n | w_{n-1})$$

כלומר, רק למילה האחורונה יש השפעה על החיזוי של המילה הבאה, וכל המילים שלפנייה הן חסרות השפעה על ההתפלגות של המילה הנחוצה (וממילא גם על המשך המשפט). באופן כללי, מודל **n-grams** מניח את המשווואה הבאה:

$$P(w_n | w_{n-1}, w_{n-2}, \dots, w_1) = P(w_n | w_{n-1}, w_{n-2}, \dots, w_{n-N}), N \leq n$$

ניקח לדוגמה מספר משפטים וננתן אותם בשיטת **bigram**:

- I know you
- I am happy
- I do not know Jonathan

נניח ונרצה לבדוק את ההסתברות שהמילה **Jonathan** היא המילה הבאה אחרי הסדרה **I do not know**. ראשית נגידיר את המילון, המכיל את כל המילים האפשריות בשפה:

$$V = \{I, know, you, am, happy, do, not, Jonathan\}$$

כעת נוכל להעיר את ההסתברות לכך על ידי ספירת כמהות הפעמים שהצמד $\langle know | Jonathan \rangle$ מופיע בטקסט, ולבראם בכמהות הפעמים ש-won't מופיע בטקסט עם מילה כלשהי (כולל הפעמים שמוופיע עם Jon). באופן פורמלי גדריר את המשוואה הבאה:

$$P(w_n | w_{n-1}) = \frac{C(w_{n-1} w_n)}{\sum_{w \in V} C(w_{n-1} w_n)}$$

כאשר האות C מסמנת את מספר הפעמים שעמדו מסוים בטקסט. ניתן לשים לב שהביטוי במכנה למעשה סופר את כמהות הפעמים w_{n-1} קיים בטקסט, וכך נוכל לפשט את המשוואה האחורונה ורשום במקומה:

$$P(w_n | w_{n-1}) = \frac{C(w_{n-1} w_n)}{C(w_{n-1})}$$

אם כך, ההסתברות לכך שהמילה Jonathan תופיע אחרי המילה know היא:

$$P(Jonathan | know) = \frac{1}{2}$$

באופן דומה ניתן לשער את ההסתברות של המילה הבאה על סמך יותר מילה אחת אחרת, למשל לדוגמה 2 מילים אחרת. מודל זה נקרא trigram, וכי שנאמר לעיל, באופן כללי מודל המסתמן על $1 - N$ מילים לצורך חישוב ההסתברות של המילה הבאה בטקסט נקרא N -gram.

המודל המركובי סובל ממספר בעיות:

1. טבלת ההסתברויות המתבקשת מאוד דיללה. כיוון שה-corpus (שהוא סט האימון) הינו בגודל מוגבל, לעיתים לא נוכל לראות את כל הקומבינציות ה אפשריות, מה שוביל להערכתה של הסתברות 0 עבור צמידים שלא מופיעים בטקסט האימון.

2. כאשר נרצה לחזות בעזרת מודל זה את ההסתברויות על טקסט חדש, כנראה שנתקל בambilים שלא נתקלנו בהן בטקסט האימון וכן לא נוכל להגיד את הסתברותה עבור N -Grams המכילים מיללים אלו.

שביל להתחמיך עם בעיות אלו – באופן מלא או חלק – נוכל להפעיל שיטות החלקה (Smoothing) על ההסתברויות של צמידים שהופיעו מעט/כליל לא הופיעו בטקסט האימון. שיטות החלקה במחנות לוקחות קצר מסת הסתברות מהארוחים השכיחים (כלומר, N -Grams המופיעים הכי הרבה) ויתורומות" לאיורומים פחות שכיחים. ישן מגוון שיטות להחלהת הסתברות ונקורו עתה חלק מהן.

Laplace Smoothing

הדרך פשוטה ביותר לבצע החלקה היא להוסיף מספר קבוע \mathcal{K} לכל האירוחים. באופן זה אנו דואגים לתת משמעות גם ליצירופים דידים שמספרם מועט של פעמים (או אפילו לא מופיעים בכלל). אם למשל יש צירוף שופיע רק פעם אחת ולשעתה יש צירוף שופיע 1000 פעמים, אז הוספה הקבוע \mathcal{K} ממשוערת שעכשו נמנא את הצירוף הראשון בכזה המופיע $(\mathcal{K} + 1)$ פעמים וביחס לציירוף השני נתיחס בכזה שהופיע $(\mathcal{K} + 1000 + 1)$ פעמים. הוספה זו כמובן ונינה משפיעה על הצירוף השני, אך היא יכולה לפחות פי כמה את הסתברותה של הציירוף הראשון. כמובן שכאשר אנחנו מתעסקים עם הסתברויות נרצה לאחר הוספה הקבוע במונח לתקן גם את מקדם הנרמול. באופן פורמלי, החלקת הסתברות נוגדרת באופן הבא:

$$P_{Laplace}(w_n | w_{n-1}) = P_{Add-\mathcal{K}}(w_n, w_{n-1}) = \frac{C(w_{n-1} w_n) + \mathcal{K}}{\sum_w C(w_{n-1} w_n) + \mathcal{K}} = \frac{C(w_{n-1} w_n) + \mathcal{K}}{C(w_{n-1}) + \mathcal{K} \cdot |V|}$$

כאשר $|V|$ הוא גודל המילון (כמה המילים המופיעות ב-corpus). היתרון בשיטה זאת היא האפסות שלה, אך עם זאת יש לה חסוך בולט הנבע מכך שהיא משתמשת באופן מושגית את ההסתברויות של מאורעות תדירים ובכך בעצם משבשת את ההסתברויות של מלמדות מהtekst. פעולה יואיה מדי מוצמת הסתברות עברת מהארוחות השכיחים למאורעות עם הסתברות נמוכה. השיטה הבאה מנסה לתקן בדיק את זה.

כמובן שניתן לבחור כל ערך \mathcal{K} שרצים ואוטו להוסיף למcka. באופן זה כן ניתן לשנות ברמה מסוימת עד כמה להגדיל את ההסתברויות לקומבינציות שאינן מופיעות בסט האימון על חשבו הקומבינציות השכיחות (כתילות- \mathcal{K}). בחירה למשל של $\mathcal{K} = 0.5$ מאפשרת למזער את העייפות היכל להתקבל משינוי הספירה.

Backoff and Interpolation

שיטת החלוקת בסעיף הקודם נותנת פתרון לקביעת הסתברות של מאוועות המקבלים הסתברות 0, וישן גישות נוספת שנותנת מענה למוגבלות האלה. נניח ואנחנו משתמשים במודל trigram, מודל המניח שהסתברות למילה הבאה תליה בשתי המילים שבאותה פניה. אם נבע את השער של כל שלישיה לפי הגדרה (=מספרת המועדים שלהם בטקסט), כל שלישית מילים שאינה מופיעה כרצף בטקסט תקבל הסתברות 0, ובעצם תקיים את המשוואה:

$$P(w_n | w_{n-1}, w_{n-2}) = 0$$

בכדי להימנע מלחת הסתברות 0 ביצה מצב, ניתן להיעזר גם בהסתברויות של bigram ו-unigram:

$$P(w_n | w_{n-1}), P(w_n)$$

שיטת back-off מציעה לחת את כל המקרים בהם קיבלו 0 ולשער אותם מחדש באמצעות מודל bigram. למעשה ניקח את כל המילים שעדיין נותרו עם הסתברות 0 (כלומר, צרכי מילים שאין מופיעים בטקסט) ונשער אותם באמצעות unigram. באופן זה מקווים להציג לפחות מספר המילים בעלי הסתברות 0 היא קטנה (עד אפסית), כיוון ששימוש במודלים יותר פשוטים מאפשר שימוש במילון רחוב יותר של קומבינציות הקיימות בטקסט. שיטה דומה נקראת Interpolation, ובזה במקומ לחת את הרצפים בעלי הסתברות 0 ולשער אותם במילון הנמצא בדרגה אחת מהותית (למשל – לשער בעררת bigram במקומות trigram), המודל מארש מתייחס לקומבינציה כלשהי של ה-words.

10.1.2 Word representation (Vectors) and Word Embeddings

עד כה, המילים השונות היו מייצגות בעדרת אוטואות. כך למשל, המילה כלביות ייצוג על ידי צירוף האותיות DOG בודע שהמילה חתול תיזג על ידי הצירוף CAT. "צוג זה מכיל מאפיינים סינטקטיים (=תחביריים) של השפה, קרי אין המילה נכתבת. עם זאת, "צוג זה חסר מאוד, כיון שהוא יתקשה בלימידת "צוג של מאפיינים סמנטיים. דוגמא להבנה סמנטית של שפה היא ההבנה שכלב וחתול הן לא מילים נרדפות, אך הן כן קשורות אחת לשנייה באופן כלשהו – שתיהן מייצגות חייה מחמד שאנשים מגדלים בדירותם. מודל המבוסס על "צוג טקסטואלי של השפה לא יכול להציג להבנה סמנטית שלה, ולכן נרצה שהייצוג שלנו יהיה מספק עשיר ויכל הן מאפיינים תחביריים והן מאפיינים סמנטיים של השפה.

בפועל, מכל למפתות את הייצוג הטקסטואלי לייצוג נומירי בצורה פשוטה בעדרת One-Hot vectors – מערך באורך המילון שלנו, המציג כל מילה במילון בעדרת 1 באיבור המתאים במערך. לדוגמה נתון המילון הבא:

Index	Word
0	Dog
1	Cat
2	Lion

ונכל לייצג את המילים השונות בעדרת וקטורים באורך 3, באופן הבא:

$$\text{Dog} \rightarrow [1, 0, 0]$$

$$\text{Cat} \rightarrow [0, 1, 0]$$

$$\text{Lion} \rightarrow [0, 0, 1]$$

כך, לכל מילה יהיה "צוג" וקטורי ייחודי. עם זאת, "צוג פשטי זה הינו בעתיי מכיוון שהוא קשה ללמידה ממאפיינים סמנטיים. כדי להבין את הסיבה לכך ראשית יש להגדיר את מושג הדמיון בעולם של וקטורים.

Cosine similarity

מעבר לייצוג וקטורי של מילים דרוש מיאתנו להגדיר דמיון בין וקטורים במרחב שנוצר. אחת מההגדרות הפופולריות לדמיון בין וקטורים היא Cosine similarity. כמו רוב השיטות לחישוב דמיון בין וקטורים, גם פונקציית דמיון זו מבוססת על מכפלה פנימית של וקטורים. נניח ונתוננו 2 וקטורים w , v , שניהם בעלי אותו הממד N . המכפלה הפנימית (dot product) בין הווקטורים האלה מוגדרת באופן הבא:

$$v \cdot w = v^T w = \sum_{i=1}^N v_i w_i$$

באמצעות הגדרה זו נוסה לאמוד את הדמיון בין היצוג של המילים כלב וחתול במרחב הווקטורי שנוצר בעקבות ייצוג בעדרת One-Hot vectors. כאמור, היצוג של המילה חתול במרחב זה הוא: $[0, 1, 0]$ → Cat, בעוד שהייצוג של המילה כלב באותו מרחב הינה: $[1, 0, 0]$ → Dog. לכן, لكن המכפלה הפנימית במרחב זה תהיה:

$$[0, 1, 0] \cdot [1, 0, 0] = 0 \cdot 1 + 1 \cdot 0 + 0 \cdot 0 = 0$$

נותראה זו מדגימה את הבנייניות בייצוג פשוטי זה, מכיוון שהיא רצויים שווקטוריהם אלה כן יהיו דומים במובן מסוים, ולא שהחוצאה תהיה 0, המלמדת על כך ככל אין קשר בין שתי המילים. למעשה בשיטה זו הדמיון בין ייצוגים של כל זוג מילים יהיה אפס.

Bag-of-words

דרך אחרת לייצר קשר בין מילים בעלות קשר סמנטי היא להתייחס לא רק למילים בזדדות אלא גם להקשרים בתוך המשפט עצמו. באופן זה נכל להגדיר ייצוג וקטורי של מילה על ידי ספירה של כמות הפעמים של מילה אחרת נמצאת אינה בהאנו ההקשר. שיטה זאת נקראת Bag-of-Words, ופנוי שונכל להדגים אותה, נציג לך מהו הקשר של מילה. באופן פשוט הקשר של מילה זה המשפט בו היא מופיעה, אך ניתן גם לחקח רק חלון של מספר מילים מתוך המשפט (לרוב אורך החלון קטן מאורך המשפט). לדוגמה, נניח ונתונם לנו הטקסט והmillion הבאים:

טקסט:

- [The dog is a domestic mammal, not wild mammal], is a domesticated descendant of the wolf, characterized by an upturning tail.
- [The cat is a domestic species of small mammal], It is the only domesticated species in the family.

million:

1. The	5. Mammal	9. Animal	13. Tail
2. Is	6. Not	10. Descendant	14. Cat
3. A	7. Natural	11. Dog	15. Species
4. Domestic	8. Wild	12. Wolf	16. Small

כעת נרצה לייצג את המילים Dog ו-Cat בשיטת Bag-of-words (=כמות המילים לפני ואחרי המילה שרוצה לייצג. חלון זה מסומן בטקסט בסוגרים מרובעתות בבעד אדום). בונה מטריצה עם מספר עמדות כאורך המילון ומספר שורות כמספר המילים אותן נרצה לייצג (לרוב מספר השורות יהיה כמספר המילים במילון), אך לשם הדוגמא נציג כאן טבלה קטנה יותר). עבור כל מילה, נבדוק כמה פעמים היא נמצאת בטקסט באופןו חלון יחיד עם מילים אחרות:

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Dog	1	1	1	1	2	1	1	1	0	0	0	0	0	0	0	0
Cat	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	0

כעת נוכל לראות שהמכפלה הפנימית בין שני הווקטורים המציגים את המילים Dog ו-Cat אינה 0. עם זאת, ישנו שתי בעיות נוספת הנובעת מפשטות פתרון זה:

1. מילים פחות משמעותיות כמו a, is, INCLUDED בספרה למרות שהן לא בהכרח מושיפות מידע ממשמעותי לייצוג.
2. מילים המופיעות בטקסט לעיתים רוחקות ידי בעלות ייצוג וקטורי מאד דיליל, מה שמאגדיל שוב את הסיכוי למכפלה פנימית בעלת ערך קטן מאוד (עד אפס) עם רוב הווקטורים האחרים.

TF-IDF

הדרך הטבעית לפתרון של הבעיה הראשונה – רעש שנוצר ממילים בעלות תדירות גבוהה שאינן תורמות ביצוג, היא סיוון של המילים האלה מהມילון. אולם, פתרון זה אינו יעיל, כיון שהתדרות של מילים משתנה בין תחומיים/דומיניים שマהם נלקח הטעט. לכן פותחה שיטת ייצוג הנקראט $TF-IDF$, ומטרתה להפחית את רעש זה באופן אוטומטי.

בשיטת $TF-IDF$, מגדירים פונקציה tf אחורית בעלת שני רכיבים. במקום להסתכל על חלון יחיד מסביב לכל מילה, ניתן להסתכל על כל המילים בתחום הנוצרים מסביב למליה המיצגת, ולתת לה ניקוד לפי התדרות של אותה מילה. באופן פורמלי, tf מוגדר בצורה הבאה:

$$tf = \log(C(w, c) + 1)$$

משמעות הביטוי היא שאנו סופרים כמה המילה c מופיעה בקונטקסט (=חלון) של המילה המייצגת w (על התוצאה מפעלים \log בשביל rescaling של ערכים גבוהים מאוד).

עד כה השיטה אינה שונה במעטה מספריה כמו שראינו בwords, Bag-of-words, אך מה שעשו את $TF-IDF$ שונה הוא הביטוי השני. הביטוי idf מוגדר בצורה הבאה:

$$df = \text{count}(w \in \text{Context})$$

$$idf = \log\left(\frac{N}{df}\right)$$

המונח df מייצג את כמות הפעמים שהמילה w מופיעה בקונטקסטים אחרים, בעוד N מייצג את כמות המילים במילון. נשים לב שגם מילה מסוימת, למשל *the*, מופיעה בכל הקונטקסטים של כל המילים במילון, אך הביטוי בתוך הלוג יהיה 1 ולכן idf יהיה 0. לעומת זאת, אם מילה מסוימת מפועעת אר וריך בקונטקסט אחד, אז הערך שהוא בתוך הלוג יהיה N .

לבסוף, $TF-IDF$ מוגדר באופן הבא:

$$TF - IDF = tf \cdot idf$$

מדד משקלל זה מציין עבור ייצוג של מילה מסוימת לתת משקל דוחי למילים אחרות הנמצאות אותה בקונטקסט באופן תדריר אך אין נמצאות בקונטקסט של מילים אחרות.

PPMI - Positive Pointwise Mutual Information

cut נרצה לפתרו את הבעיה השנייה – בעית הייצוג הדليل של מילים שאין תדירות בטקסט. שיטת PPMI מגדירה פונקציית ניקוד המחשבת את היחס בין הסיכוי של שתי המילים להמצאה יחד לעומת הנטה בונפרד – $\frac{P(x,y)}{P(x)P(y)}$. cut בשביל לחשב את הערך של התא המייצג את המילה y בוקטור הייצוג של המילה x השתמש בהסתברות הנ"ל ונפעריל לוג. אם ההסתברות לראות את המילים x, y ביחד לכפוף להסתברויות לראות כל אחת בלבד, נקבל שערך הביטוי הוא $\log(1)$ כלומר 0. לעומת זאת, אם הסיכוי לראות את המילים הללו ביחד יותר מאשר היחידים x, y נקבע ערך הגדול מ-1. ישנו מקרה נוסף, בו הסיכוי לראות את הביטויים בחודד גודל מסוימי שיראו אותם יחד. במקרה זה הביטוי שנקבל היה קwon מ-1 וכן תלוג יהילה שליל, אך מכיוון שהערכיהם שליליים נזדים להיות לא אמינים אלא אם הטקסט שללו גודל מסוים). בנוסף עד אלמנט קון לפונקציית החישוב:

$$PPMI = \max\left(\log\frac{P(x,y)}{P(x)P(y)}, 0\right)$$

בפועל זה יוכל לנормיל את הערך הנמוך עבור מילים נדירות בטקסט.

Word2Vec

השיטות שראינו עד כה לחישוב וקטורי הייצוג של המילים אפשרות לנו רקוד מאפיינים סטטיסטיים בייצוג המילים. עם זאת, יש כמה חסרונות לשיטות אלו: ראשית, הן יוצרות וקטורים מאד דילילים, ובנוסף לכך גודל הווקטורים תלוי בכמות המילים שיש להם במילון, מה שיוצר וקטורים גדולים שמכבידים על החישובים במשימות השפה השונות. למשל – ראיינו קודם שណיתן ליציג מילה באמצעות וקטור שכלו אפסים למעט תא אחד עם הערך 1 במיון ייחודי לכל מילה. עברו שפה עם אלפי מילים ואף יותר מכך, כל וקטור המיציג מילה הוא באורך עצום, ועם זאת הוא מאד דיליל כיוון

שיש בו רק מספר נאים מועט שערכם שונה מ-0. לכן, נרצה לפתח שיטה שתיצור וקטורי ייצוג דחוסים (dense) בעלי ממד קטן יותר.

שיטת Word2Vec הינה שיטת Self-Supervised שפותחה למטרת יצרה של וקטורי ייצוג דחוסים של מיללים. הפרידגמה של למידה מונחית (supervised learning) ריאלה, אך התוויות אינם נתונים אלא נוצרם באופן אוטומטי מתוך הדאטה הלא מתוויות. באופן זה ניתן לאמן מודלים עם כמות גודלה של דאטה לא מותיא בצורה 'עהלה ולא צורך בתיאוג' (상을 להיות מאוד יקר). בהקשר זה, אלגוריתם Word2Vec משמש בא-Self-supervision באופן של Self-Sampling-With-Negative-Sampling – SGNS. הרעיון מאחורי גישה זו הוא להגדיר עלייה סיווג שptrתת להזנת מה הוסתרה של מיללים שונים להוות לוגיקות בkontekst של מילה נתונה. כדי האימון לוחקים את המושגים שיצורו בעקבות תהליכי אימון המשימה הראשית, והם הוו הייצוג של המילה. ב כדי לבן זאת לעומק, נבחן טקסט פשוט יחסית בעזרת אלגוריתם Word2Vec. נניח ונתן המשפט הבא כחלק מהtekst האימון שלנו:

Folklore, legends, myths, and fairy tales have followed childhood through the ages.

ראשית נקבע את אורך החלון (=מספר המילים עליה מסתכמים בסביבות כל מילה) – 3.Cut נקבע על הטקסט וניצור תיוגים בין כל מילה במילון ליתר המילים. למשל עבור המילה tales נסoxic לדאטה סט שלם דוגמאות חיבובות של המילים שנמצאות בkontekst עם tales. בנוסף, כדי למנוע התנומות של כל היצוגים לוקטור בודד, נctrar "להראות" למודל איך נראות דוגמאות שליליות ולכך להשתמש בשיטה הנקראת negative sampling. בשיטה זו דוגמים מהמילון בהסתברות פרופורציונלית לתדרות המילה (עם תיקון קטע קצר יותר סיכוי למילים נדירים) את המילים שישמשו אותנו כדוגמאות שליליות. הרעיון אחוחוי תהליך זה הוא שאפשר יש לנו מיליון גדול המילים שנגזרו לא להיות קשורות למילה שעבורה אנחנו יוצרים את הדוגמאות שליליות.

Folklore, legends, [myths and fairy [tales] have followed childhood] through the ages.

word	context	Label
tales	myths	+
tales	and	+
tales	fairy	+
tales	have	+
tales	followed	+
tales	childhood	+
tales	great	-
tales	April	-
tales	the	-
tales	young	-
tales	orphan	-
tales	dishes	-

ככל לייצר דאטה מותיא עבור משימת הסיווג, וככל להשתמש בתיוגים אלה בשbill לאמן את המודל. הכוונה במשמעות סיווג בהקשר זה מעט שונה מסיווג במובן הפשוט של המילה: מטרת המודל היא שבחינתן מילה (במקרה שלנו tales), נרצה שהייצוג של מילים המופיעות באותו kontekst עם כל מילה (במקרה שלנו – אלו שופיעות עם

"tales" באותו חלון) יהיה קרוב לייצוג של tales בעוד שמיילים שאינן מופיעות באותו קונטיקסט יהיו בעלות ייצוג "שונה" (mbhinitiy מכפלה פינית או cosine similarity). נניח שהברנו שהייצוג של כל מילה יהיה וקטור בגודל 100. נניח את הותלייר c של מילה מקבלת וקטורי רדום. סמן את וקטורי הייצוג של המילה w ב- e_w ואת וקטורי של מילת kontekst- c . השאייה היא שהמכפלה הפינית של וקטורי הייצוג של המילים בkontekst של w יהיה גבוהה יותר, בעוד שהמכפלה הפינית של וקטורי ייצוג של מילים שאינן מופיעות באותו kontekst יהיה נמוך. כאמור עליל, Cosine similarity (המטריקה המגדירה דמיון בין וקטורים) היא בעצם מכפלת פנימית של הווקטורים (=מכפלת dot עם נרמול). כתת מכל להגדיר בעית logistic regression באופן הבא:

$$p(+|w, c) = \sigma(e_w \cdot e_c) = \frac{1}{1 + e^{-e_w \cdot e_c}}$$

$$p(+|w, c) = 1 - p(+|w, c)$$

ובהתאם לכך פונקציית המטרה (Loss) תוגדר באופן הבא:

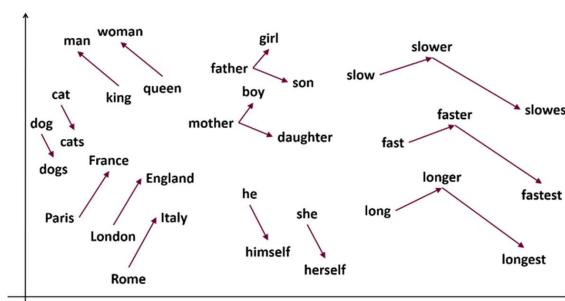
$$L = -\log[p(+|w, c_{pos}) \prod_{i=1}^k p(-|w, c_{neg_i})]$$

$$= -[\log(p(+|w, c_{pos})) + \log(\prod_{i=1}^k p(-|w, c_{neg_i}))]$$

$$= -\left[\log(\sigma(e_w \cdot e_{c_{pos}})) + \sum_{i=1}^k \log(1 - \sigma(e_w \cdot e_{c_{neg_i}})) \right]$$

שים לב שאנו מניחים אי תלות בייצוג של הדוגמאות השליות. בכך שנבצע מענייניזציה לפונקציית מטרה זו נגרם למכפלה הפינית בין וקטורי של מילה לבן מילת הקונטיקסט להוות גבואה וכן בזמן למכפלה הפינית בין וקטורים שאינם בkontekst להוות נמוכה. כך הייצוג של המילה "tales" (קרוב להזמה "domeh") ל"יצוג של המילים שאינם בkontekst. את תהליכי המנייניזציה המשך האימון נוכל לבצע באמצעות stochastic gradient descent.

אחד התוצאות הפוטות והחשבונות של שיטת Word2Vec ניתנת להמחשה על ידי פריסת וקטורי הייצוג במדוד נמוך. בעזרת שיטות מתקדמות להורדת ממד (כפי שהוסבר בהרחבה בפרק 2), ניתן ליצור בדו-ממד או תלת ממד את וקטורי הייצוג של המילים לאחר האימון.



איור 11.2 וקטורי הייצוג של מילים לאחר ביצוע embedding word2vec באמצעות דומה מייצגים על ידי וקטורים באותו כיוון.

נוכל לבדוק אם המרכיב הווקטורי מקודד מאפיינים סמנטיים. לדוגמה, ניתן לראות שהווקטור המחבר בין וקטורי הייצוג של המילים King, Man מקביל ובעל אורך דומה לווקטור המחבר בין הייצוג של Queen, Woman –Queen, Woman נספה – הווקטור בין שם של ארכ' לעיר הבירה שלו מקביל ובעל אורך דומה לווקטור שבין ארץ אחרת ועיר הבירה המתאימה. Queen, Woman סמנטיים, כמו למשל שייחסו בין King, Man.

10.1.3 Contextual Embeddings

מנגנון בנייה ייצוגי מילים (embedding) שראינו עד כה למדו ייצוג סטטי עבור כל מילה. אך דבר זה יכול להיות בעייתי מכיוון שפה טبيعית היא דינמית ותלויה בקשר, ולאויה מילים יכולות להיות כמה פירושים. בשבל להבין את הבעייתיות נסתכל על המשפטים הבאים:

1. We need to **book** the flight as soon as possible

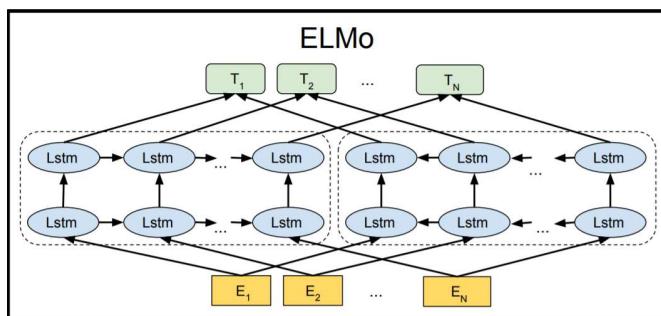
2. I read the **book** already

למילה **book** יש לפחות שתי משמעויות במשמעותם האלו. במשמעות הראשון המילה משמשת כפועל ובמשמעות השני כשם עצם עם תפקיד סמנטי שונה במשמעות. אם כך ברור שההקשר שבו המילה מופיעה משפיע על המשמעות שלה אך מגנוני embedding כמו **word2vec** יציג את המילה באותו וקטור יציג עבורה שני המופיעים.

לכן נרצה לפתח מנגנון embedding עבור המילים, שבאמצעותו וקטור המיציג מילה יהיה תלוי בהקשר בו היא מופיעה.

Embeddings from Language Models (ELMo)

אחד השיטות הראשונות שהציגה טכניקה למידת יציגות למילים הינה Embeddings from Language Models, או בקיצור SLM – ארכיטקטורה הבונה לכל מילה יציגותabh בהקשר שלה בתוך המשפט. הרעיון במודול זה הוא לחקות יציגות של מילה, להוסיף לו מידע נוסף מההקשר של המילה במשפט ולקבל יציגות חדש התלוי גם בהקשר שלה. בניסוח אחר ניתן לומר ש-SLM הינה פונקציה המקבלת משפט שבו כל מילה מיצגת שבדרך כלל השניה (למשל –Word2Vec), ומושגיה ליציגות של המילה בתוך כל המשפט. פעולה זו נעשתה על ידי מיפוי אימון מודל שפה דו-כיווני – המודול לומד לחזות גם את המילה הבאה בטקסט וגם את המילה הקודמת, ובכך הוא לומד לתת למילה גם את ההקשר שלה. ארכיטקטורתו הרותה רוראית כר:



איור 11.3 ארכיטקטורת ELMo. הקלט הינו משפט המיציג כל מילה קיבלה מידע נוספת על ההקשר שלה וכעת מיצגת באופן חדש. מהלך האימון והוספת ההקשר בין המילים נעשו באמצעות שכבות של ריבי LSTM.

כפי שמתואר בפרק 6.2.1, כל בלוק של LSTM מקבל קלט לשני רכיבים המציגים את ההיסטוריה של המשפט עד הנקודה בה מופיעה המילה של הרגע הנוכחי (c_t, h_t), וקלט נוסף של האיבר הנוכחי בסדרה, שבמקרה שלנו זה המילה הנוכחיית (e_t). המוצא של ה-LSTM הינו יציג חדש המשקלן את רכיבי ההיסטוריה יחד עם הייצוג הנוכחי של המילה.

בדומה לשיטות אחרות ליצירת יציגות יציגו למילים, אנחנו מאמנים את המודול בעזרת משימת מידול שפה וחוזים את המילה הבאה בהינתן המילים הקודמות. אך בשונה מאלגוריתמים אחרים, ELMo משתמש בארכיטקטורתה דו-כיוונית, כך שבתהליך האימון מושלבת משימת שפה נוספת הנוספת המונזה את המילה הקודמת בהינתן הסוף של המשפט. הארכיטקטורתה של ELMo מבוססת שכבות של LSTM שומרוכבות זו על גבי זו, ולפי כתבי המאמר השכבות התחנות מצליחות ללמידה פיזית (למשל מאפיינים סינטקטיים למיניהם), בעוד שהשכבות העליונות לומדות פיצ'רים מורכבים (למשל מאפיינים סמנטיים, כמו משמעות המילה בהקשר).

לאחר תהילך האימון ניתן להקפייה את הרכਮטרים של המודול ולהשתמש בו עבור משימות אחרות. הכותבים מציעים לששרר את הייצוג הווקטורי של LSTM בכל שכבה כהה שייכל אינטגרציה גם מתחילה המשפט עד המילה הנבדקת וגם מסוף המשפט עד המילה הנבדקת. מה שקרה בפועל זה שהשכבות החבויות (hidden layers) של LSTM הם עצם מהווים את יציגי המילים בשיטת יציגות Chat, כלומר כל מילה במשפט מיצגת על ידי הטא המקיים בשכבה ה-LSTM השעליה. בנוסף הם מושגים מספר פרמטרים קון שמאפשר כויל (Fine tune) עבור משימה ספציפית. כך לדוגמא יוכל להתאים את הייצוג של המילים למשימת סיווג של יענות משימתית תיאוג של ישותות במשפט.

פה חשוב להדגש נקודת מרכזית – בסופו של דבר התוצר של ELMo הינו **מודול שפה הולך יציג של טקסט** והוא יציג **תלוי הקשר**. שכבות ה-LSTM השונותammenות מודול שפה על מנת לייצר יציג עבורה המילים,

המתיחס גם להקשר. לאחר סיום האימון של מודל השפה (pre-training), ניתן לנקח אותו ולבצע learning transfer, כלומר להשתמש ביצוגים שהוא מפיק גם לשימוש אחרות על ידי הוספת שכבות בקצה. לאחר פרסום המאמר, Sebastian Ruder (חוקר NLP מפורסם) טען כי:

"It is very likely that in a year's time NLP practitioners will download pretrained language models rather than pre-trained word embeddings"

כלומר, כעת מי שירצה לבצע משימת שפה כבר לא יתבוסס רק על ייצוג סטטי של המילים אלא הוא יסתמך על מודל שפה מאומן שיודיעו לנקח ייצוג התחלתי של מילום ולהפוך אותו ליצוגים קונטקטואליים. ס.ELMo והוא מאמרים רבים אחרים אימנו מודל שפה מאומנים שיניתם לנקח אותו ולהשתמש בהם עבור משימות קיצה שונות על ידי הוספה של כמה שכבות וכיול המודל.

Bidirectional Encoder Representations from Transformers (BERT)

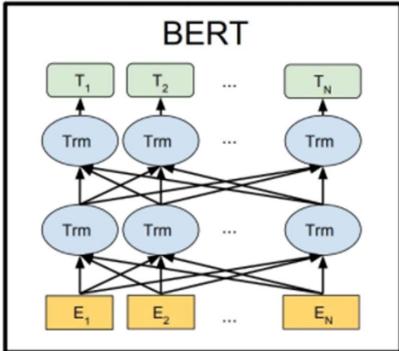
כאמור, הרעיון של ELMo הוא לייצר contextual embedding, ובכך לקבל מודל שפה המאפשר לנקח ייצוג וkontextual של מילים ולהעיר אותו במידע על ההקשר של כל מילה בטקסט. לмерות ש-ELMo משתמש בשני היכיונים של המשפט (חוזה את המשך המשפט מתחילה ואת תחילתו המשפט מסווף) הוא אין למדashi היכיונים בתהילר אחד, אלא צריך לחלק את הלמדיין לשני חלקים שונים. במקרה זה השהසר בתקדמה פארק, 8, מעתה מעתסדים עם סדרות ארוכות של מילים, וקטור היצוג של כל איבר נהיה בעיתוי כיוון שהוא מוגבל ביכולת שלו להכיל קשרים בין מספר רב של איברים. בambilים אחרות, כאשר רציתם להוציאו לתקטורו היצוג של מילה מסוימת קשור למילוי רוחוקות, אנו מאלצים את היצוג "לזכור" מידע רב, אך היצוג הווקטורי של המילים ב-ELMo אינו מצליח לעשות זאת בaczורה מסווג טובה. לכן, על אף הצלחת גישה זו במשימות שונות, היא עדין התקשה במשימות בהן נדרשת יכולת לנתח טקסטים ארוכים (כמו למשל משימה של summarization). בKİוסף לכך, האלגוריתם יחסית איטי, כיוון שככל פעם הוא מסתכל על מילה אחת בלבד.

בדי להתמודד עם בעיות אלו וליצור ייצוג המסוגל להכיל מידע איקוטי גם ברצפים ארוכים, ניתן להשתמש ב-self attention (מסובב בהחבה בפרק 8). אחד השימושים הרשומים במנגנון self-attention עיבוד שפה היה בטרנספורמרים, ובפרט בארכיטקטורת רשת הנקראט BERT , המבוססת על ה- BERT encoder של הטרנספורמර המקורי. שימוש זה היה פרייצת דורך בתחום, וכךים ברוב המוטול של המהקר והיפויו בתחום ה- NLP -משתמשים בארכיטקטורת רשת מבוססת attention. למעשה, BERT מציע שיטה לבניית ייצוג קונטקטואלי של מילים הבאה להתמודד עם החולשות הקיימות ב- MoSE . נתאר בקצרה את העקרונות של מגנון self-attention, attention ב-self attention:

באופן הכל פושט, בהקשר של עיבוד שפה self-attention הוא מנגן שמשמעותו את הקשרים של כל מילה בטקסט כלשהו ביחס לשאר המילים באותו טקסט. כאשר מבצעים self-attention על קטע טקסט, מקבלים ייצוגים דדיטים של המילים הלאקティים בחישובם בס את הגדירים בין המילים השונות באותו טקסט, בaczות אפוי של מגנון self-attention. ניתן לבנות ייצוג של מילה שתali בקשרים שלא בהכרח מוצאים בקטע טקסט, ככלומר הקשרים המתקבלים בין המילים – כלים להיזות מיזגים בזורה טובה גם עבר רציפים ומילים שאינן נמצאות בסמכות יחסית (שכמו זה היה אחד החסרונות הגדולים של ELMo). בKİוסף, מגנון זה מיותר את הוצרך לעבר מילה אחר מילה בקטע טקסט לצורך בניית ייצוג המילים שבו. במקום מעבר זה, ה- BERT -encoder מקבל הקלט את כל קטע הטקסט כמקרה אחד, מה שעשוי להקטין את הזמן הדרוש לבניית היצוג של המילים. אך ה- BERT -encoder בטרנספורמר יכול לשמש מודל שפה, אם מאמנים אותו צורה מותאמת.

בשונה ממודל LSTM ישומר את המצב בכל נקודת זמן ובזמנים מוקודד את המיקום של כל מילה בערך שהקלט מתקיים כמילה בזאת בכל פעם, מודל הטרנספורמר מקבל את כל הקלט בבתacha. לכן בשביל לנקח בחשבון את המיקום של כל מילה במשפט אנחנו משתמשים באלמנון נוסף שנקרא Positional Embedding. אלמנון זה מקובד וקטור ייחודי לכל מיקום במשפט ובתוך ממצאים יהיבור של הווקטור שנוצר מהמיוקום.

המפתחים של BERT ימצאו מארכיטקטורת הטרנספורמר המקורי את ה- BERT -encoder, והגידו רישימת אימון חדשה בכך להפוך אותו למודל שפה. בכך לבנות מודל מוצלח, תהילר האימון של BERT כל שתי משלימות: 1. Masked Language Model (MLM) – באופן רנדומלי עושים masking למילים מסוימות, ומטרת המודל הוא לחזות את המילים החסרות. 2. Next Sentence Prediction (NSP) – המודל מקבל קלט זוגות של משפטים מקטע טקסט, ומטרת המודל היא לחזות האם המשפט השני הוא המשך המשפט הראשון בSAMPLE. ארכיטקטורת הרשת נראה כך:



איור 11.4 ארכיטקטורת BERT. הקלט הינו משפט המיצג כלשהו, והפלט הוא אותו משפט אך כל מילה קיבלה מידע נוסף על ההקשר שלו וכעת מייצגת באופן חדש. תהליך האימון והוספת ההקשר בין המילים נעשו באמצעות self-attention operation.

גם BERT, בדומה ל-ELMo, מציע בסופו של דבר מודל שפה מאומן הייעד לקחית טקסט המיצג באופן מסוים ולהוציא לו מידע על היחס בין המילים השונות שבtekst. תהליך יצירת המודל היה אמנים ייר, אך כעת ניתן לחקת אותו ייחסית בקלות לכלי אותו ואף להוציא שכבות בקצח עברו מושיות שפה שונות.

GPT: Generative Pre-trained Transformer

עם הכניסה של מנגנון-h-attention וטרנספורמרים לעולם-NLP, הוצעו יותר ויותר מודלים שפה מבוססי attention. לצורך ההמחשה ניתן לציין שבשים הבזוזות שעברו מАЗיא BERT, הוא צוטט כבר בעשרות אלפי מאמרים. אחד המודלים הייתר מפוזרמים הינו Generative Pre-Training (GPT). מודל GPT-הינו מודל שעבוד בשיטת auto-regression, כלומר, כאשר המודל חוזה את המילה הבאה הוא מוסיף את המילה לקלט עבור האיתריזה הבאה. כך הוא יכול ליצור מושפעים מהתחילה של מילה בודדת. אם נרצה לד-יק, המודלים הללו לא תמיד משתמשים במילים ייחודה האטומית, לעיתים אנחנו נעבד עם חלקי-מילים ואפיו אוטואטיים להם נקרא טוקנים או איסימוני. דבר זה יכול לעזור לנו בהכללה ולהקטים את הסיסי לטוקן שלא נמצא במילון (Out of Vocabulary).

הארQUITטורה של GPT הבנוי מ-Transformers מה שמאפשר לבנות ארכיטקטורה عمוקה שמתבססת בקונטקסט של המשפט עבור כל מילה (Contextual embeddings). ארכיטקטורת הינה היחידה המרכזית של GPT, כאשר בשונה מ-BERT-GPT משתמש רק ב-decoder (מנגן-h-attention) שמקודד את הפיצרים, והפלט שלו הינו הטוקן הבא.

השכבה הראושנה בארכיטקטורה של GPT היא שכבה הנקראת Input encoding והוא הופכת את המילים (או ליותר דיווק הטוקנים) לוקטורים, כלומר הוא מבצעת word embedding.

לאחר קידוד הקלט נשתמש במודל-h-Transformer כדי לקודד פיצרים מהם נסיק את הטוקן הבא. התהליך הזה מתבצע באמצעות רכיב הנקרא Masked Self attention. בשונה ממנגנון self-attention כל טוקן בזירת הקונטקסט של כל שאר הטקסט, GPT צריך לקודד כל טוקן ורק בעזרת הטוקנים שכבר קדמו לו, כיוון שהשלב זה המידע היידי שקיים זה הטוקנים שנמצאו עד כה (וכמובן שאנו גישה לטוקנים שעדיין לא נוצרו). כאשר מקודדים את הייצוג עבור טוקן מסוים, רכיב Masked Self attention מספק כל וקטור של טוקן שבא אחריו, כך שהמודל לא יכול למודוד יצוג התלו依 מילים שבאות לאחר הטוקן המופיע, אלא עליו להפיק את המירב מהתוקנים הקודמים לו.

כיוון-Sh-auto-regressive פועל בצורה של GPT, ניתן לאחר אימון ליצור טקסט באמצעותו – ניתן לモודל התחלת קצירה של טקסט, ובכך ממנו ליצור את המילים הבאות. כך בכל שלב ניתן לו קלט את הטקסט הראשוני ואת הטוקנים שייצר בשלבים הקודמים, והוא ימשיך וייצר עוד ועוד טקסט.

Perplexity

לאחר בניית מודל שפה, נרצה "למדו" עד כמה הוא מוצלח. לצורך זה יש להגדיר מטריקה מתאימה. המטריקה היכי נפוצה למדידת "עוצמה" של מודל שפה הינה perplexity, שהיא מושג הלוקו מהתורת האינפורמציה והוא מודד כמה טוב מודל השפה חוזה את השפה ב-Corpus שאותוניסינו למודל.

לפוי' שנסביר את המושג באופן פורמלי' ניתן אינטואיציה למה אנו מצפים לקבל מהamodelika השובחר. נניח ואנו מבצעים את הפעולה הבאה: ראשית לוקרים משפט שלם ווותאים ממנו את התהילה ומכוונים כkill' למודל שפה ומבקשים מהמודל לחזות את המשפט המשפט. כמובן שנרצה לקבל חיזוי שדומה כל האפשר לשפט המקורי, וכן למדוד הצלחה של מודל על ידי השוואת הפלט שלו לשפט האמיתי. באופן יותר ככל' ניתן לקחת טקסט המכיל כמה משפטים כראצנו, להיכנס חלים ממנה למודל השפה, ולהשוו את הפלט המתתקבל לטקסט המקורי. כיוון שמודל שפה הינו הסתברותי, השוואת הפלט לשפט המקורי באופן מילוי בלבד אינה מספקת, כיוון שהיא אינה משקפת בצורה מספקת טובות מידה ההצלחה שלו. אם למשת בשפט המקורי היה תהה היליה "לבנה" ואילו המודל חזה את המילה "ירח", השוואת שתי המילים כשלעצמם מראה לאורה שהמודל שגה חלוטין, אך בפועל לנו ידועים שניים בלבד לנו לדפנות לכך הפלט של המודל ביחסן חלק מהמשפט המקורי. מודד המסוגל לבחון עד כמה סביר לקביל את הפלט של המודל ביחסן חלק מהמשפט המקורי.

מדד perplexity בא להתמודד עםאתגר זה, והוא אכן פועל בצורה שונה מהאופן בו תיארנו את ההשווואה הפשטית בין טקסט המקורי לבין הפלט של מודל השפה. מדד זה מסתכל רק על הטקסט המקורי, והוא עובר מילה-מילה בטקסט זה ובודק מה ההסתברות שמודל השפה ינבא את המילה הבאה בטקסט בהינתן כל המילים שלפניה. ככל שההסתברות יותר גבוהה, כך המודל יותר מוצלח. באופן פורמלי', perplexity מוגדר באופן הבא:

$$PP(W) = P(w_1, w_2, \dots, w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1, w_2, \dots, w_N)}}$$

כל שמודל מנבא בהסתברות גבוהה יותר את המילים של המשפט המקורי, כך המוניה שבתוך השורש יהיה יותר גדול, ומילא כל הביטוי עצמו של perplexity. המדד קיטן יותר. לעומת זאת, ככל שערך perplexity קיטן יותר, כך המודל מוצלח יותר. נפתח מעט את הביטוי האחרון בעדרת כל השרשות:

$$= \sqrt[N]{\frac{1}{\prod_{i=1}^N P(w_i | w_{i-1}, w_{i-2}, \dots, w_1)}}$$

למשל עבור מודל מבוסס bigram, המדד יהיה פשוט יותר וראה כך:

$$= \sqrt[N]{\frac{1}{\prod_{i=1}^N P(w_i | w_{i-1})}}$$

כאמור לעיל, ככל שערך המדד perplexity נמוך יותר, כך מודל השפה איכותי יותר.

10. References

<http://d2l.ai/>

ELMo, BERT:

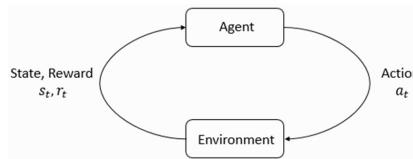
<https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html>

11. Reinforcement Learning (RL)

רוב האלגוריתמים של עולם הלמידה הינם מבוססי DATA, כלומר, בהינתן מידע מסוים הם מנסים למצואו בו חוקיות מסוימת, ועל בסיסו לבנות מודל שיכל להתאים למקרים נוספים. אלגוריתמים אלה מחולקים לשניים:

1. אלגוריתמים של למידה מונחית, המבוססים על דאטה $\{x_i = y_i\}_{i=1}^n \in \mathbb{R}^{d \times d}$, כאשר $x_i \in \mathbb{R}^d$ הוא אוסף של אובייקטים (למשל נקודות במרחב, אוסף של תמונות וכדו), ו- $y_i \in \mathbb{R}^d$ הוא אוסף של labels. לכל אובייקט $x_i \in \mathbb{R}^d$ יש מותאים $y_i \in \mathbb{R}^d$.
2. אלגוריתמים של למידה לא-מונחית בעורם הדאטא $\{x_i\}_{i=1}^n \in \mathbb{R}^d$ הוא אוסף של אובייקטים ללא labels, ומנסים למצאו כלים מסוימים לעדעתה זה (למשל – חלקה לאשכולות, הורדת מדדי).

למידה מבוססת חיזוקים הינה פרדיגמה נוספת תחת התחום של למידה מכונה, כאשר במקורה זה הלמידה לא מסתמכת על דאטה קיימ, אלא על חקירה של הסביבה ומציאת המדיניות/הסטרטגייה הטובה ביותר. שוטון שנמצא בסביבה שאינה מוכנה, וعليו לבצע עדדים קר שהתגמל המctrller אותו הוא יקבל מהסקימ. בלמידה מבוססת חיזוקים, בוגר לפרידגיות האחרת של למידה מכונה, השיטה לא דושה מבעוד מועד. השוטון נמצוא באירועים וឌוות ואינו יודע בשום שלב מה הצעד העכשווי, אלא הוא רק מקבל פידבק על הצעדים שלו, ורק הוא לומד מה כדי לעשות ומה כדי להימנע. באופן כללי ניתן לומר שטורת הלמידה היא ליצור אסטרטגייה קר שבלכל מני מצבים לא ידועים השוטון יבחר בפעולות שבאופן מctrller ימייעילות עבורו. נתאר את תהליך הלמידה באופן גרפי:



איור 11.1 מודל של שוטון וסביבה.

בכל צעד השוטון נמצא במצב s_t ובוחר פעולה a_t המעבירו אותו למצב s_{t+1} , בהתאם לכך הוא מקבל מהסביבה תגמול r_t . האופן בו מוצבצעת הלמידה היא בעדרת התגמול, כאשר נרצה שהשוטון יבצע פעולות המוצאות אותו בתגמול חיובי (חיזוק) וימנע מפעולות עברו הוא מקבל תגמול שלילי, ובמctrller הוא ימќסם את כל התגמולים עברו כל הצעדים שהוא בחר לעשות. כדי להבין כיצד האלגוריתמים של למידה מבוססת חיזוקים עבדים ראשית יש להגדיר את המושגים השונים, ובנוסף יש לנתח באופן פורמלי את התיאור המתמטי של תהליכי הבעה השונים.

11.1 Introduction to RL

בפרק זה נגדר באופן פורמלי תהליכי מרכוב, בעזרתם ניתן לתאר בעיות של למידה מבוססת חיזוקים, ונראה כיצד ניתן למצוא אופטימום לביעות אל בהינתן מודל וכל הפרמטרים שלו. לאחר מכן בקשרו במספר שיטות המנסות למצוא אסטרטגיה אופטימלית עבור תהליכי מרכוב כאשר לא כל הפרמטרים של המודל נתונים, ופרקם הබאים נדבר על שיטות אלה בהרחבה. שיטות אלה הן למשה הלב של למידה מבוססת חיזוקים, כיון שהן מוסות לחזז אסטרטגיה אופטימלית על בסיס תגמולים ללא ידיעת הפרמטרים של המודל המרכוב עבורי ו齊ים למצוא אופטימום.

11.1.1 Markov Decision Process (MDP) and RL

המודל המתמטי העיקרי עליו בניין אלגוריתמים השונים של RL הינו תהליכי החלטה מרכוב, כלומר תהליכי שבה המעברים בין המצבים מקיימים את תכונת מרכוב, לפיה ההסתפלות של מצב מסוים תלויה רק במצב הקודם לו:

$$P(s_{t+1} = j | s_1, \dots, s_t) = P(s_{t+1} = j | s_t)$$

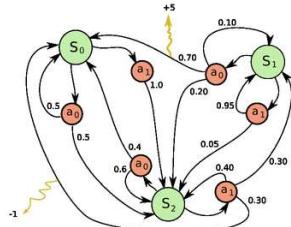
תהליכי החלטות מרכובי מתוארים על ידי סט הפרמטרים $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}\}$:

- State space (\mathcal{S}) – מרחב המצבים של המערכת. המצב ההתחלתי מסומן ב- s_0 .
- Action space (\mathcal{A}) – מרחב הפעולות. s הוא מרחב הפעולות האפשריות במצב S .
- Transition (\mathcal{T}) – הביטוי: $[0, 1] \rightarrow \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ הינו פונקציית מעבר, המחשבת את ההסתברות לעבור בזמן t במצב s_t למצב s_{t+1} על ידי הפעולה a : $P(s'_{t+1} = s' | s_t = s, a_t = a) = \mathcal{T}(s' | s, a)$.
- במשמעותו: $P(s'_{t+1} = s' | s_t = s, a_t = a) = \mathcal{T}(s' | s, a) = \mathcal{P}(s_{t+1} = s' | s_t = s, a_t = a) = \mathcal{P}(s_{t+1} = s' | s, a)$.
- מעשה מיצגת את המודול – מה ההסתברות שבחירת הפעולה a במצב s תביא את השוטון למצב s' .
- Reward (\mathcal{R}) – הביטוי: $\mathbb{R} \rightarrow \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ הינו פונקציה הננתנת תגמול/רווח לכל פעולה a הגורמת למעבר ממצב s למצב s' , כאשר בדרך כלל $\mathcal{R}_a \in [0, 1]$. לעיתים מסומנים את התגמול של הצעד בזמן t ב- r_t .

המרקוביות של התהיליך באה לידי ביטוי בכר שמצב s מכיל בתוכו את כל המידע הנחוץ בכך לקבל החלטה לגבי a_t , או במלים אחרות – כל ההיסטוריה עצמה שמורה בתוך המצב s_t .

ריצה של MDP מאופיינת על ידי הריבועית הסודורה $\{s_t, a_t, r_t, s_{t+1}\}$ – פעולה a_t המתרחשת בזמן t וגורמת למעבר ממצב s_t למצב s_{t+1} , ובנוסף מקבלת תגמול מיידי r_t , כאשר $r_t \sim \mathcal{R}(s_t, a_t)$.

מסלול (trajectory) הינו סט של שלשות $\{s_t, r_t, a_t\}$, $t = 0, 1, 2, \dots$, כאשר המצב ההתחלתי מוגדר מהתפלגות כלשהיא ($s_0 \sim p_0(\cdot)$), והמעבר בין הממצבים יכול להיות דטרמיניסטי $s_{t+1} = f(s_t, a_t)$ או סטוכסטי $s_{t+1} \sim p(\cdot | s_t, a_t)$.



איור 11.2 תהליך קבלת החלטות מركבי. ישנו שלושה מצבים – $\{s_0, s_1, s_2\}$, ובכל אחד מהם יש שתי פעולות אפשריות (עם הסתברויות מעבר שונות) – $\{a_0, a_1\}$. עבור חלק מהפעולות יש תגמול שונה מ-0. מסלול יהיה מעבר על אוסף של מצבים דרך אוסף של פעולות, שליל אחד מן שייגרם.

אסטרטגיה של סוכן, המסתמנת ב-π, יוננה בבחירה של אוסף מחללים. בבעיות של למידה מבוססת חיזוקים, רצחה למצוא אסטרטגיה אופטימלית (Optimal Policy) $\pi: S \rightarrow A$: ממקסיממת את התגמול המוצע $\mathbb{E}[\mathcal{R}(s_t, \pi(s_t))]$. כיוון שלא תמיד אפשר לחשב באופן ישיר את האסטרטגיה האופטימלית, ניתן להגדיר ערך החזרה (Return) המבטא סכום של תגמולים, ומנסים לקסם את התוחלת של $\mathbb{E}[R|s, \pi]$. ערך החזרה נקבע באמצעות החזרה הנוכחית (discount return) $\gamma \in (0, 1]$, והוא מוגדר באופן הבא: עבור פרמטר $\gamma \in (0, 1]$, הינו הסכום הבא:

$$Return = \sum_{t=1}^T \gamma^{t-1} r_t$$

אם $\gamma = 1$, אז מעתניינים רק בתגמול המיידי, וככל ש- γ גדול כך יותר נתונים יותר משמעות לתגמולים עתידיים. כיוון ש- $\mathbb{E}[r_t | s, \pi] = \frac{1}{1-\gamma}$, הסכום חסום על ידי $\frac{1}{1-\gamma}$.

התוחלת של ערך החזרה נקראת **Value function**, והוא נותנת לכל מצב ערך מסוים המשקיף את תוחלת התגמולו שנייה להישג דרך מצב זה. באופן פורמלי, כאשר מתחילה במצב s , $\mathbb{E}[R|s, \pi]$ מוגדר להיות:

$$\mathcal{V}^\pi(s) = \mathbb{E}[R(\tau)|s_0 = s]$$

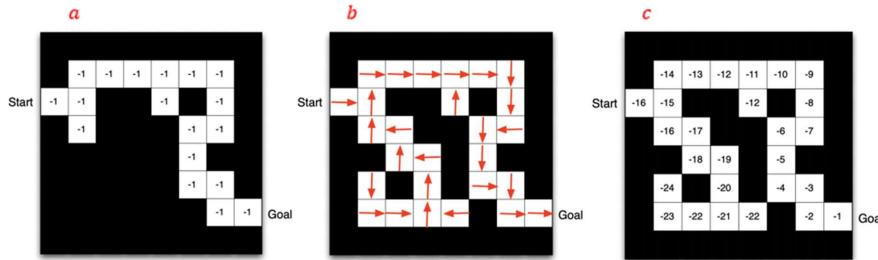
בעזרת ביטוי זה ניתן לחשב את האסטרטגיה האופטימלית, כאשר ניתן לנוקט בגישה ישירה ובגישה עקיפה. הגישה הישירה מנסה למצוא בכל מצב מה הפעולה הכי נכונה. בהתאם לכך, חישוב האסטרטגיה האופטימלית עשויה באופן הבא:

$$\pi(s) = \arg \max_a \sum_{s'} p_a(s, s') (\mathcal{R}_a(s, s') + \gamma \mathcal{V}^\pi(s'))$$

לעתים החישוב הישיר מסובך, כיוון שהוא צריך ללקח בחשבון את כל הפעולות האפשריות, ולכן מסתכנים רק על Value function. לאחר שלכל מצב יש ערך מסוים, בכל מצב הסוכן יעבור למצב בעל הערך הכי גדול מבן כל הממצבים האפשריים אליום ניתן לעבור. חישוב הערך של כל מצב נעשה באופן הבא:

$$\mathcal{V}(s) = \sum_{s'} p_\pi(s, s') (\mathcal{R}_\pi(s, s') + \gamma V(s'))$$

ניתן לשין לב שבעוד הגישה הראשונה מתמקדת במציאת **אסטרטגיה/מדיניות אופטימלית** על בסיס הפעולות האפשריות בכל מצב, הגישה השנייה לא מסתכלת על הפעולות אלא על הערך של כל מצב, המשקף את תוחלת התגמול שניתן להציג כאשר נמצאים במצב זה.



איור 11.3 (a) מודל: המשחק של הטוקן והוא המשבצת בו הוא נמצא, הפעולות האפשריות הן ארבעת הכוונים, כל פעולה גוררת תגמול של -1 , והסיכוייות העבריו קבועות לפני המצבים של המשבצות (א) אפשר לילכט למשבצות שחומות. (b) מדיניות – החלטה בכל מצב איזה צעד לבצע. (c) Value של כל משבצת.

לסיום, ניתן לומר שכל התהום של RL מבוסס על שלוש אבני יסוד:

מודל: האופן בו אנו מתארים את מרחב המצבים והפעולות. המודל יכול להיות נתון או שנוצר לשלב אחר אותו, והוא מורכב מהסתברויות מעבר בין מצבים ותגמול עבור כל צעד:

$$P_{ss'}^a = p_\pi(s, s') = P(s_{t+1} = s' | s_t = s, a_t = a)$$

$$\mathcal{R}_{ss'}^a = \mathcal{R}_\pi(s, s') = \mathbb{E}[r_{t+1} | s_t = s, a_t = a, s_{t+1} = s']$$

- פונקציית המדיניות את התוחלת של התגמולים העתידיים:
 $\mathcal{V}^\pi(s) = \mathbb{E}[R(\tau) | s_0 = s]$

מדיניות/אסטרטגיה – בחירה (Deterministic or Stochastic) של צעד בכל מצב נתון:
 $\pi(s|a)$

11.1.2 Bellman Equation

לאחר שהגדכנו את המטרה של למידה מבוססת חיזוקים, ניתן לדבר על שיטות לחישוב אסטרטגיה אופטימלית. בפרק זה נתייחס ל מקרה הספציפי בו נתון מודל מركובי עם כל הפורטטים שלו, כלומר אוסף המצבים, הפעולות והסתברויות המעבר ידועים. כאמור, פונקציית התוחלת של ערך ההחזרה עבור אסטרטגיה נתונה π , כאשר מתחילה במצב s :

$$\mathcal{V}^\pi(s) = \mathbb{E}[R(\tau) | s_0 = s] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \right]$$

ביטוי זה מסתכל על הערך של כל מצב, בלי להתייחס לפועלות המבעירות את הסוכן במצב אחד למצב אחר. נתינו ערך לכל מצב יכול לסייע במציאת אסטרטגיה אופטימלית, כיוון שהוא מדרגת את המצבים השונים של המודל. באופן דומה, ניתן להגיד את ה-Value function – Action-Value function של ערך ההחזרה עבור אסטרטגיה נתונה π , כאשר במצב s מבצעים את פעולה a , ולאחר מכן ממשיכים לפי האסטרטגיה π :

$$\mathcal{Q}^\pi(s, a) = \mathbb{E}[R(\tau) | s_0 = s, a_0 = a] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = t \right]$$

ביטוי זה מסתכל על הזוג (s_t, a_t) , כלומר בכל מצב יש התייחסות למצב הנוכחי ולפעולות האפשריות במצב זה. בדומה ל-Value function, גם ביטוי זה יכול לסייע במציאת אסטרטגיה אופטימלית, כיוון שהוא מדרג עבור כל מצב את הפעולות האפשריות.

ונכל לסמן ב- $\mathcal{V}^*(s)$ ו- $\mathcal{Q}^*(s, a)$ את הערכים של האסטרטגיה האופטימלית π^* . Optimal Action-Value function

$$\mathcal{V}^*(s) = \max_{\pi} \mathbb{E}[R(\tau)|s_0 = s], \mathcal{Q}^*(s, a) = \max_{\pi} \mathbb{E}[R(\tau)|s_0 = s, a_0 = a]$$

הרבה פעמים מתעניינים ביחס שבין \mathcal{V} -ו- \mathcal{Q} , ויתן להיעדר במערכות הabeiים:

$$\mathcal{V}^\pi(s) = \mathbb{E}[\mathcal{Q}^\pi(s, a)]$$

$$\mathcal{V}^*(s) = \max_{\pi} \mathcal{Q}^*(s, a)$$

באופן קומפקטי ניתן לרשום את (s^*, \mathcal{V}^*) כך:

$$\forall s \in S \quad \mathcal{V}^*(s) = \max_{\pi} \mathcal{V}^\pi(s)$$

כלומר, האסטרטגיה π^* הינה האופטימלית עבור כל מצב s .

icut נתון מודל מרקובי עם כל הפרמטרים שלו – אוסף המצבים והפעולות, הסתברויות המעבר והתגמול עבור כל פעולה, ומעוניינים למצאו דרך פעולה אופטימלית עבור מודול זהה. ניתן לשועז זאת בשתי דרכים עיקריות – מציאת האסטרטגיה $(\pi^*|s)$ האופטימלית, או חישוב Value-e-un \mathcal{V}^π של כל מצב ובחרת מצבים בהתאם לערך זה. משימות אלו יכולות להיות מסובכות מאוד עבור מושגים מורכבים וגדולים, ולכן לעיתים קרובות משתמשים בשיטות איטרטיביות ובקיורובים על מנת לדעת כיצד להנוג בכל מצב. הדרכ הפשוטה לחישוב (s^*, \mathcal{V}^π) משתמשת ב-Bellman equation על תכונות דינמי. נפתח את הביטוי של (s^*, \mathcal{V}^π) מתוך ההדרה שלו:

$$\mathcal{V}^\pi(s) = \mathbb{E}[R(\tau)|s_0 = s] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \right]$$

נפתח את הסכום שבתוכלו לשני איברים – האיבר הראשוני יתר האיברים:

$$= \mathbb{E}_\pi \left[r_{t+1} + \gamma \cdot \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s \right]$$

icut נשמש בהגדרת התוחלת ונקבל:

$$\begin{aligned} &= \sum_{a,s} \pi(a|s) p_\pi(s, s') \left(\mathcal{R}_\pi(s, s') + \gamma \cdot \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s \right] \right) \\ &= \sum_{a,s'} \pi(a|s) p_\pi(s, s') (\mathcal{R}_\pi(s, s') + \gamma \cdot \mathcal{V}^\pi(s')) \end{aligned}$$

הביטוי המתקיים הוא מערכת משוואות לינאריות הניתנות לפתרון באופן אנלטי, אם כי סיבוכיות החישוב יקרה. נסמן:

$$V = [V_1, \dots, V_n]^T, R = [r_1, \dots, r_n]^T$$

$$T = \begin{pmatrix} p_{11} & \cdots & p_{1n} \\ \vdots & \ddots & \vdots \\ p_{n1} & \cdots & p_{nn} \end{pmatrix}$$

ונקבל משווהה מטריצונית:

$$V = R + \gamma T V \rightarrow V = R + \gamma T V$$

$$\rightarrow \mathcal{V}^\pi(s) = (\mathbb{I}_n - \gamma T)^{-1} R$$

בגלל שהעריכים העצמיים של T חסומים על ידי 1, בהכרח יהיה ניתן להפוך את $T - \gamma \mathbb{I}_n$ מה שmbטיח שייה פתרון למשווהה, ופתרון זה הוא אף יחיד עבור \mathcal{V}^π . כמשמעותם את V ניתן למצוא גם את \mathcal{Q}^π על ידי הקשה:

$$\mathcal{Q}^\pi(s, a) = \sum_{s'} p_\pi(s, s') (\mathcal{R}_\pi(s, s') + \gamma \mathcal{V}^\pi(s')) = \sum_{s'} p_\pi(s, s') \left(\mathcal{R}_\pi(s, s') + \gamma \sum_{a'} \pi(a'|s') \mathcal{Q}^\pi(a'|s') \right)$$

Iterative Policy Evaluation

הסיבוכיות של היפוך מטריצה הינו $(n^3)\mathcal{O}$, ועבור א' גדול החישוב נהיה מאוד יקר ולא 'יעיל'. כדי לחשב את הפתרון באופן 'יעיל', ניתן לומר להשתמש בשיטות איטרטיביות. שיטות אלו מבוססות על אופרטור בילמן, המוגדר באופן הבא:

$$BO(V) = R^\pi + \gamma T^\pi \cdot V$$

ניתן להוכיח שאופרטור זה הינו העתקה מכווצת (contractive mapping), כלומר הוא מקיים את התנאי:

$$\forall x, y: \|f(x) - f(y)\| < \gamma \|x - y\| \text{ for } 0 < \gamma < 1$$

במילים: עבור שני וקטורים במרחב, אופרטור $f(\cdot)$ ומספר γ החסום בין 0 ל-1, אם נפעיל את האופרטור על כל אחד מהווקטורים ונחשב את נורמת ההפרש, נקבל מספר קטן יותר מאשר הנורמה בין הווקטורים כולל הפקטור γ . אופרטורו המקיים תכונה זו הינו העתקה מכווצת, כיוון שנורמת ההפרש של האופרטור על שני וקטורים קטנה מnorמת ההפרש בין הווקטורים עצם. הוכח:

$$\|f(u) - f(v)\|_\infty = \|R^\pi + \gamma T^\pi \cdot u - (R^\pi + \gamma T^\pi \cdot v)\|_\infty = \|\gamma T^\pi(u - v)\|_\infty$$

$$\text{מטריקת אינסוף מוגדרת לפ': } \|s(s) - v\|_\infty = \max_{s \in S} |u(s) - v(s)|.$$

$$\|\gamma T^\pi(u - v)\|_\infty \leq \|\gamma T^\pi\|_\infty \|u - v\|_\infty$$

הביטוי $\|\gamma T^\pi\|_\infty$ למעשה סוכם את כל ערכי מטריצת המעברים, שכן הוא מסתכם ל-1, ונקבל:

$$= \gamma \|u - v\|_\infty$$

ובכך הוכחנו את הדרוש.

לפי משפט נקודת השבת של בנර, להעתקה מכווצת יש נקודת שבת (fixed point) יחידה המקיימת $x = f(x)$ וזרה $(x_t)_{t+1} = f(x_t)$ המתכנסת לאוֹת נקודת שבת. שכן נוכל להשתמש באלגוריתם איטרטיבי עבור γ שיביא אותנו לנקודות שבת, ולפי המשפט זהוי נקודת השבת היחידה ומילא הגענו להתכנסות. בפועל, נשתמש באלגוריתם האיטרטיבי הבא:

$$V_{k+1} = BO(V_k) = R^\pi + \gamma T^\pi \cdot V_k$$

נסתכל על הדוגמא הבאה:

$$T^\pi = \begin{pmatrix} 0.8 & 0.1 & 0.1 & 0 & 0 \\ 0.1 & 0.8 & 0.1 & 0 & 0 \\ 0 & 0.1 & 0.8 & 0.1 & 0 \\ 0 & 0 & 0.1 & 0.8 & 0.1 \\ 0 & 0 & 0.1 & 0.1 & 0.8 \end{pmatrix}, R^\pi = \begin{pmatrix} 0.1 \\ 1.3 \\ 3.4 \\ 1.9 \\ 0.4 \end{pmatrix}, \gamma = 0.9$$

באמצעות השיטה האיטרטיבית נקבל:

$$V_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, V_1 = \begin{pmatrix} 0.1 \\ 1.3 \\ 3.4 \\ 1.9 \\ 0.4 \end{pmatrix}, V_2 = \begin{pmatrix} 0.6 \\ 2.6 \\ 6.1 \\ 3.7 \\ 1.2 \end{pmatrix}, \dots, V_{10} = \begin{pmatrix} 7.6 \\ 10.8 \\ 18.2 \\ 16.0 \\ 9.8 \end{pmatrix}, \dots, V_{50} = \begin{pmatrix} 14.5 \\ 17.1 \\ 26.4 \\ 26.8 \\ 18.4 \end{pmatrix}, V^\pi = \begin{pmatrix} 14.7 \\ 17.9 \\ 26.6 \\ 27.1 \\ 18.7 \end{pmatrix}$$

ניתן לראות שאחרי 50 איטרציות הפתרון המתקובל בצורה האיטרטיבית קרוב מאוד לפתרון המתקובל בצורה האנליטית.

Policy Iteration (PI)

חישוב-value function מאפשר לנו לחשב את ערכו של $(s)^\pi \mathcal{U}$ עבור כל s , אך הוא אינו מבטיח שנגוע לאסטרטגיה האופטימלית. נניח והצלהנו לחשב את $(s)^\pi \mathcal{U}$ וממנו אנו יודעים לגזר אסטרטגיה, עדין יתכן שקיימת פעולה a שייתר משתלימת מארש הפעולה המומצעת לפי האסטרטגיה הנגזרת מ- $(s)^\pi \mathcal{U}$. באופן פורמלי ניתן לתאר זאת בצורה פשוטה – נניח שוחישבנו את $(s)^\pi \mathcal{U}$ ואת $(a)^\pi \mathcal{U}$ יתכן וקיימת פעולה עבורה:

for such s, a : $\mathcal{Q}^\pi(s, a) > \mathcal{V}^\pi(s)$

אם קיימת פעולה כזו, אז ישתלם לבחור בה ולאחר מכן לחזור לפועל בהתאם לאסטרטגיה $(s|a)\pi$ הנקראת מחייבת Value function. למעשה, ניתן לחפש את כל הפעולות עבורן כדי לבצע פעולה מסוימת עבורה התגמול יהיה גבוה יותר מאשר האסטרטגיה של $(s)\pi'$. באופן פורמלי יותר, נרצה להגדיר אסטרטגיה דטרמיניסטיית, עבורה בהסתברות 1 ננקוט בכל מצב s ב פעולה היכי קדאית: a :

$$\pi'(a|s) = 1 \text{ for } a = \arg \max_{a'} \mathcal{Q}^\pi(s, a')$$

נשים לב שרעיון זה הוא בעצם להשתמש באסטרטגיה גרידית – בכל מצב ננקוט ב פעולה היכי משתלמת בטעו של צעד ייחיד, ואז להמשיך עם האסטרטגיה הנתונה. השאללה העולה היא כמובן – מדוע זה בהכרח נכון? כמובן, האם הרעיון שאומר שלא משנה באיזה מצב אנו נמצאים, הבחירה של הפעולה האופטימלית בהכרח תוביל לקבלה אסטרטגיה יותר טובה מאשר האסטרטגיה הנוכחית? בכך לוחכית זאת ננסה כמפורט:

בהתברות 2 אסטרטגייות π' , π , כאשר π' דטרמיניסטית, אז כאשר $(s|\pi') > (\mathcal{V}^\pi(s, \pi'))$ בהכרח לכל s יתקיים: $(s|\pi') > (\mathcal{V}^\pi(s))$. ראשית נפתח לפי הגדרה:

$$\mathcal{V}^\pi(s) < \mathcal{Q}^\pi(s, \pi'(s)) = \mathbb{E}_\pi[r_{t+1} + \gamma \cdot \mathcal{V}^\pi(s_{t+1}) | s_t = s, a_t = \pi'(s)]$$

כיון שהאסטרטגיה הינה דטרמיניסטית, הפעולה הבוחרת אינה רנדומלית ביחס ל- π' , ולכן נוכל לרשום:

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma \cdot \mathcal{V}^\pi(s_{t+1}) | s_t = s]$$

עת לפיו אותו אי שוויון שבנהה מכל לבצע את אותן חישובים לעוד הבא s_{t+2} :

$$< \mathbb{E}_{\pi'}[r_{t+1} + \gamma \cdot \mathcal{Q}^\pi(s_{t+1}, \pi'(s_{t+1})) | s_t = s]$$

זה שוב שווה לו:

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma \cdot r_{t+2} + \gamma^2 \cdot \mathcal{V}^\pi(s_{t+2}) | s_t = s]$$

וכך הלאה, ולמעשה הוכחנו את הדורש – נקיטת הפעולה היכי עילה בכל מצב תמיד תהיה יותר טובה מהਪתרון של $\mathcal{V}^\pi(s)$.

עת יש לבדוק שני טכניקות שאנו יודעים לבצע:

Evaluation (E) – בהינתן אסטרטגיה מסוימת מכל לפתור את משוואות בלמן ולקבל את $(s|\mathcal{V}^\pi)$

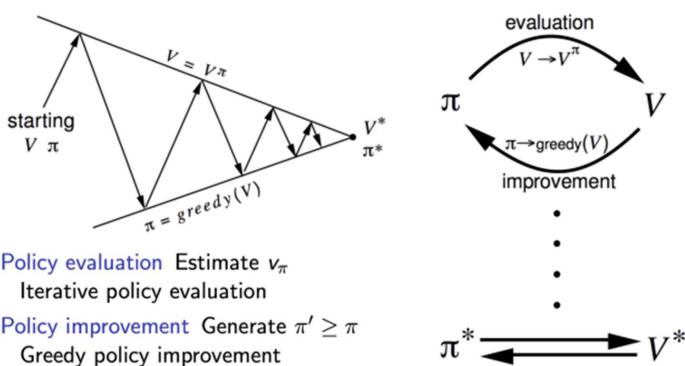
Improve (I) – בהינתן הערך של Value function, מכל לשפר אותה באמצעות בחירה גרידית של פעולה.

בעזרת טכניקות אלו ניתן להתחילה אסטרטגיה רנדומלית, ואז לבצע איטרציות המורכבות ממשתי הטכניקות האלה באופן הבא:

$$\pi_0 \xrightarrow{E} \mathcal{V}^{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} \mathcal{V}^{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \dots$$

תהליך זה נקרא Policy iteration – בכל צעד בו יש לנו אסטרטגיה נתונה עבורה משוואות בלמן ובכך נחשב את ה-Value function שלה, ולאחר מכן נשפר את האסטרטגיה באמצעות policy improvement, שכך מוביל מבצע בחירה אירידית שבטוווה הקצר טוביה יותר מאשר ה-Value function שחויבנו. ניתן להוכיח שאחרי מספר סופי של איטרציות האסטרטגיה ת收敛 לנקודת שבט (fixed point), ואז הפעולה הבאה ל-PI האסטרטגיה תהיה דобра לבחירה הגרידית:

$$\pi(s) = \arg \max_a \mathcal{Q}^\pi(s, a) = \pi'(s)$$



איור 11.4 – ביצוע איטרציות של Policy iteration ו-Policy evaluation. על מנת למצוא בכל שלב את ה-value function ולשפר אותו באמצעות ביריה גרידית.

Bellman optimality equations

השלב הבא בשימוש ב-*policy iteration* הוא **להוכיח** שהאסטרטגיה אליה מתכוונים הינה אופטימלית. נסמן את נקודת השבתה ב- π^* ונקבל את הקשר הבא:

$$V^{\pi^*}(s) \equiv \mathcal{V}^*(s) = \max_a Q^*(s, a) = \max_a \sum_{s'} p_\pi(s, s') (\mathcal{R}_\pi(s, s') + \gamma \cdot \mathcal{V}^*(s'))$$

ובאופן דומה:

$$Q^*(s, a) = \sum_{s'} p_\pi(s, s') (\mathcal{R}_\pi(s, s') + \gamma \cdot \max_{a'} Q^*(s', a'))$$

משוואות אלה נקראות **Bellman optimality equation**. ניתן לשים לב שהן מודמות למשוואות בלבד מן צאמנו, אך במקומם התוחלתת שהיא תלה לנו בהתחלה, נתע יש max. נרצה להראות שהפתרון של משוואות אלה הוא ה-Value function האופטימלית. ננסח את הטענה באופן הבא:

אסטרטגייה הינה אופטימלית אם ורק אם היא מקיימת את **Bellman optimality equation**. Bellman optimality equation מגדירה את האסטרטגיה הינה אופטימלית אם היא בהכרח מקיימת את משוואות האופטימליות, כיוון שהראינו שהן מתקבלות מנקודת השבתה אליה האיטרציות מתכנסות. אם האסטרטגיה לא הייתה אופטימלית אז היה ניתן לשפר עוד את האסטרטגיה ולא היינו מגיעים עדין לנקודת השבתה. בשביל להוכיח את הclaim השני נשתמששוב ברעיון של העתקה מכוזחת. נגדיר את האופרטור הבא:

$$BV(s) = \max_a \sum_{s'} p_\pi(s, s') (\mathcal{R}_\pi(s, s') + \gamma \cdot \mathcal{V}(s))$$

ניתן להראות שאופרטור זה הינו העתקה מכוזחת, ומילא לפיה המשפט של בך יש לו נקודת שבת יחידה. כיוון שהראינו ששימוש ב-*policy iteration* מביא את האסטרטגיה לנקודת שבת מסוימת, תוכל לצרף לכך רק את העובדה שהאופרטור שהגדכנו הינו העתקה מכוזחת ומילא נקל שאותה נקודת שבת הינה יחידה, ומילא אופטימלית.

Value Iteration

הראנו שבעזרת שיטת *Policy iteration* ניתן להגיע לאסטרטגיה אופטימלית, אך התהליך יכול להיות איטי. ניתן לנוקט גם בגישה יותר ישירה ונסota לחשב באופן ישיר את הפתרון של משוואות האופטימליות בלבד (ופרפקטן הינו אופטימלי כיוון שהראינו שהפתרון הוא נקודת שבת יחידה). נתחיל עם פתרון רדומלי \mathcal{V}_0 ולאחר מכן נציב איטרציות באופן הבא עד שנגיע להתקנסות:

$$\mathcal{V}_{k+1} = \max_a \sum_{s'} p_\pi(s, s') (\mathcal{R}_\pi(s, s') + \gamma \cdot \mathcal{V}_k(s'))$$

נשים לב שבשיטת זו אין לנו מידע לגבי האסטרטגיה אלא רק חישבנו את ה-Value function, אך ממנה ניתן לגזר את \mathcal{Q} ואז לבחור באסטרטגיה גרידית, שהינה במקורה זה גם אופטימלית:

$$\pi(s) = \arg \max_a \mathcal{Q}^\pi(s, a)$$

ניתן להראות כי בשיטה זו התחכשות מהירה יותר ודרשות פחות איטרציות מהשיטה הקודמת, אך כל איטרציה יותר מורכבת.

Limitations

לשתי השיטות – Value iteration ו-Policy iteration – יש שני חסרונות מרכזיים:

1. הן דרשות לדעת את המודל והביבה באופן שלם ומדויק.

2. הן דרשות לעדכן בכל שלב את כל המצבים בו זמן. עבור מערכות עם הרבה מצבים, זה לא מעשי.

11.1.3 Learning Algorithms

בפרק הקודם הוסבר כיצד ניתן לחשב את האסטרטגייה האופטימלית וערך ההערכת בהינתן מודל מركזוי. השתמשנו בשתי הנחות עיקריות על מנת להתמודד עם הבעיה:

1. Tabular MDP – הנקנו שהביעה סופית ולא גדולה מדי, כך שנוכל ליצג אותה בזיכרון ולפטור אותה.

2. Known environment – הנקנו שהמודול ידוע לנו, כלומר נתונים לנו מטריצת המעברים שקובעת מהו ה-*reward* במצב s למצב s' כשווקטים בפועל a (סימנו את זה בתור (s', s') , ובנוסף נתנו לנו מה ה-*reward* המתקיים עבור כל action a (סימנו את זה בתור (s', s')).

בעזרת שתי הנחות פיתחנו את משוואות בלמן, כאשר הינו לנו שני צמדים של משוואות. משוואות בלמן עבור אסטרטגיה נתונה נכתבות באופן הבא:

$$\mathcal{V}^\pi(s) = \sum_{a,s'} \pi(a|s) \mathcal{P}_{ss'}^a (\mathcal{R}_{ss'}^a + \gamma \cdot \mathcal{V}^\pi(s'))$$

$$\mathcal{Q}^\pi(s, a) = \sum_{s'} \mathcal{P}_{ss'}^a \left(\mathcal{R}_{ss'}^a + \gamma \sum_{a'} \mathcal{Q}^\pi(s', a') \right)$$

ובנוסף פיתחנו את המשוואות עבור הפתרון האופטימלי:

$$\mathcal{V}^*(s) = \max_a \sum_{s'} \mathcal{P}_{ss'}^a (\mathcal{R}_{ss'}^a + \gamma \cdot \mathcal{V}^*(s'))$$

$$\mathcal{Q}^*(s, a) = \sum_{s'} \mathcal{P}_{ss'}^a \left(\mathcal{R}_{ss'}^a + \gamma \max_a \mathcal{Q}^*(s', a') \right)$$

הראינו שתי דרכים להגעה לפתרון האופטימלי:

1. Policy improvement Policy evaluation .

2. Value iteration – פתרון משוואות בלמן באופן ישיר באמצעות איטרציות על ה-Value function.

כאמור, דרכי פתרון אלו מניחים שהמודול ידוע, ובנוסף שמדובר במצבים אינטגרליים ויכול להיות מיוצג בזיכרון. האתגר האמייתי מתחילה בכך לפחות אחת מהנחות אלה אינה תקופה, ולמעשה פה מתחילה התפקיד של אלגוריתמי RL. עיקר ההתקומות של אלגוריתמים אלו יהיה למצאו באופן ייעיל את האסטרטגיה האופטימלית כאשר לא נתונם פורמליטים של המודול, ואז צריך לשער וותם (Model-based learning) או לממצאו דרך חישוב האסטרטגיה האופטימלית ללא שימוש במודול (Model free learning). אם למשל יש משחק בין משתמש לבין המחשב, אלגוריתמים מסוימים למודול-learning Model based learning ינסו ללמידה את המודול של המשחק או להשתמש במודול

קיים, ובעזרת המודל הם יונסו לבחון כיצד יgive המשמש לכל תור שהחישב 'בחירה' לעומת זאת אלגוריתמים מסווגים לא יתעניינו בכך, אלא יונסו ללמידה ישירות את האסטרטגיה הטובה ביותר עבורה המחשב.

היתרון המשמעותי של אלגוריתמים המסתמכים על המודל של הבעה (Model-based) נובע מהיכולת לתכנן מספר צעדים קדימה, כאשר עברו כל בחירה של פעללה המול בוחן את התוצאות אפשריות, את הפעולות המתאימות לכל תגובה, וכך הלאה. דוגמא מפורסמת לכך היא תוכנת המחשב AlphaZero שאמונה לשחק משחקי לחן כגן שחמט או גו. במקרים אלו המודל הוא המשחק והחוקים שלו, והתוכנה משתמשת בידע זהה בכך בחון את כל הפעולות והתוצאות למשך מספר צעדים רב ובחירה של הצעה הטובה ביותר.

עם זאת, בדרך כלל אף בשלב האימון אין לסייע חיצוני מהו הצעד הנכון באופן אוטומטי, ועליו ללמידה מהণיסין. לעומת זאת מוצאה כמזה אתגרים, כאשר העיקרי בינהם הוא הסכמה שהאסטרטגיה הנלמדת תהיה טובה רק עבור המקרים אותם ראה הטעון, אך לא תואם למקרים חדשים שיבואו. אלגוריתמים שמחפשים באופן ישר את האסטרטגיה האופטימלית אמנים לא משתמשים בידע שיכל להציג מבחנת צעדים עתידיים, אך הם הרבה יותר פשוטים למיושן ולאימון.

באופן מעט יותר פורמלי ניתן לנתח את ההבדל בין הגישותocr: גישת Model-based learning מנסה למצוא את הפרמטרים המגדירים את המודל $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}\}$ ואז בעזרתם לחשב את האסטרטגיה האופטימלית (למשל בעזרת משוואות בלמן). הגישה השנייה לעומת זאת לא מעוניינת לחשב במפורש את הפרמטרים של המודל אלא למצוא באופן ישר את האסטרטגיה האופטימלית $(\pi_t | a_t)$ שעבור כל מצב קובעת באיזה פעולה ללקוט. ההבדל בין הגישות נוגע גם לפונקציית המחיר לה נרצה למצוא אופטימום.

בכל אחד משני סוגים הלמידה יש אלגוריתמים שונים, כאשר הם נבדלים אחד מהשני בשאלת מהו האובייקט אותו מעוניינים ללמידה.

Model-free learning

בגישה זו יש שתי קטגוריות מרכזיות של אלגוריתמים:

- Policy Optimization – ניסוח האסטרטגיה כבעית אופטימיזציה של מציאת סט הפרמטרים θ הממקסם את $(s | a)_\theta$. פתרון בעיה זו יכול להיעשות באופן ישר על ידי שיטת עבורה פונקציית המחיר $(\pi_\theta | s) = \mathbb{E}[R(s)]$, או בעזרת קירוב פונקציה זו ומציאת מקסימום עבורה.
- Q-learning – שערוך $(s, a) \rightarrow Q_\theta(s, a)$. מציאת המשערך האופטימלי יכולה להתבצע על ידי חיפוש θ שיספק את השערוך הטוב ביותר ביחס למציאות, או על ידי מציאת הפולה שמתמקסם את המשערך: $a = \arg \max_a Q_\theta(s, a)$

השיטות המנסות למצוא אופטימום לאסטרטגיה הן לרבות policy-off, כלומר כל פעולה נקבעת על בסיס האסטרטגיה המעודכנת לפי הפעולה הקודמת. Q-learning – לעומת זאת הוא לרוב אלגוריתם policy-on, כלומר בכל פעולה ניתן להשתחמם בכל הידע שצבר עד כה. היתרונו של שיטות האופטימיזציה נובע מכך שניתן למצוא באופן ישר את האסטרטגיה הטובה ביותר, בעוד שאלגוריתם Q-learning רק משעריך את $(s, a)^*$, ולעתים השעריך לא מספיק הרצואה המתבקשת אינה מספיק טוביה. מצד שני, כאשר השעריך מצליח, הביצועים של Q-learning טובים יותר, כיון שהיחסimos ביחס לידע על העבר מנצח בקרה עיליה יותר מאשר באלגוריתמים המבוצעים אופטימיזיה של האסטרטגיה. שתי הגישות האלה אין דורות לחולות, וישנם אלגוריתמים שונים לשלב בין הריעיות ולנצח את החזוקות והיתרונות שיש לכל גישה.

Model-based learning

גם בגישה זו יש שתי קטגוריות מרכזיות של אלגוריתמים:

- Model-based RL with a learned model – אלגוריתמים המנסים ללמידה חן את המודל עצמו והן את ה-Value function או את האסטרטגיה π .
- Model-based RL with a known model – אלגוריתמים המנסים למצוא את ה-Value function ו/או את האסטרטגיה כאשר המודל עצמו נתון.

ההבדל בין הקטגוריות טמון באתגר אותו מנסים להתמודד. במקרים בהם המודל ידוע, המודד של אי הוזדות לא קיים, ולכן ניתן להתמקד בביטויים אסימפטומטיים. במקרים בהם המודל אינו ידוע, הדגש העיקרי הוא על למידת המודל.

11.2 Model Free Prediction

לאחר שסקרנו בפרק המבוא את הבסיס המתמטי של בעיות RL והציגו את משוואות בלמן ופתרון, בפרק הבא נציג גישות שונות להתמודדות עם בעיות RL עבורי פתרונות אלה אין מוספיים – או מפני שהמודל אינן ידוע או מפני שהוא Scale-free יותר מזה שהוא פטור באמצעות משוואות בלמן. בפרק זה נציג שתי שיטות הבאות להתמודדות עם מקרים בהם המודל אינו ידוע (לטמור האלגוריתם הימני **Model-Free**), והדרך שלהם להתמודד עם אתגר זה הינו **לשער את האסטרטגיה האופטימלית** בדרכים אחרות מאשר ידיעת המודל.

11.2.1 Monte-Carlo (MC) Policy Evaluation

האלגוריתם הראשון אותו נציג הינו Monte Carlo, והוא מציין דרך לשערר את ה-value function בלי לדעת את המודל. ראשית נסביר בקצרה מהו אלגוריתם Monte Carlo ואז נראה כיצד ניתן ליחסו בעיות RL. נניח ונרצה לשערר תוחלת של פונקציית התפלגות כלשהיא – $\mathbb{E}_p[f(x)]$. התוחלת יכולה להיות סכום או אינטגרל שקשhaft מאוד לחשב. ניתן לשערר את התוחלת על ידי דוגמאות רנדומליות מההתפלגות וחישוב הממוצע של הדוגמאות:

$$x_1, \dots, x_n \sim p(x)$$

$$\mathbb{E}_p[f(x)] \approx \frac{1}{n} \sum_i f(x_i)$$

לפי חוק המספריים האגדולים הממוצע של הדוגמאות מתכנס לתוחלת. משערר זה הינו חסר הטיה, ובנוסף השונות שלו קטנה ביחס לינארי לכמות הדוגמאות:

$$\mathbb{E}\left[\frac{1}{n} \sum_i f(x_i)\right] = \mathbb{E}[f(x)]$$

$$Var\left[\frac{1}{n} \sum_i f(x_i)\right] = \frac{Var[f(x)]}{n}$$

כאמור לעיל, ה-value function הינה תוחלת עבור אסטרטגיה נתונה π , כאשר מתחילה במצב s :

$$V^\pi(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \right]$$

אמנם התוחלת היא על סכום אינסופי, אך עברו מקרים רבים ככל הנראה שהarieaza היא סופית (Episodic MDP). בפועל נבעו את הפעולה הבאה: עברו אסטרטגיה נתונה π , ניצר ממנה ריצה של T צעדים, ואז ניקח מצב מסוים ונסתכל על כל ה-rewards שמקבלים ביריצה זו החל ממנה. באופן זה קיבלנו value עבור הערך של אותו מצב. חוזרת על אותה פעולה שוב ושוב וטייר ריצות נוספות ערכיהם שונים למצב מסוים, ומיציעו על פני הערכים יתנו שערר לערך האמיטי של אותו מצב. נעיר בהערכת אגב שכיוון שמצבים יכולות לחזור על עצם, נוצרת בעיה שבריצה ציוו המצבים אינם בלתי תלויים. בכך לhattגרבר על כך, אם מצב חוזר על עצמו יותר מפעם אחת מאפשרים לדגום רק את המופיע הראשון של אותו מצב ולא את יתר המופיעים (ישן עוד דרכים להtagגרבר על כך, אך זהה הדריך פשוטה ביותר). באופן פורמלי, נניח ושים למ' ריצה של T צעדים:

$$S_1, A_1, R_1, \dots, S_{T-1}, A_{T-1}, R_{T-1}, S_T$$

אז דגימה אחת מתוך התפלגות $p_\pi(\sum_{k=0}^{\infty} r_{t+k+1} | s_t = S_t)$ תראה באופן הבא:

$$G_t = R_{t+1} + \gamma R_{t+2} + \dots + \gamma^{T-1} R_T = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1}$$

למשל, הדגימה G_1 תהיה מרכיבת מכל ה-rewards שהגיאש לאחר הצעד הראשון: $R_1 + \gamma R_2 + \dots + \gamma^{T-1} R_T$. הסכום הזה ייתן לנו value עבור אותו מצב ממנו התחלנו (S_1), ועל ידי שערר אותו מצב שוב ושוב ביחס לריצות שונות תוכל לקבל ערכים שונים, ולאחר מכן למצוות בהם בדמיון G_1 לשערר את ה-value function של אותו מצב S_1 .

באופן פורמלי, העדכן של מצב לאחר דגימה נראה כך:

#sample of s_t : $N(S_t) = N(S_t) + 1$

$$\text{update the value of } s_t: \mathcal{V}(s_t) = \mathcal{V}(s_t) + \frac{1}{N(S_t)}(G_t - \mathcal{V}(s_t))$$

ולאחר הרבה דוגמאות השערק מתקיים על ידי התוחלת שלהן:

$$\mathcal{V}(s) = \mathbb{E}_{\pi}[G_t | s_t = s]$$

באופן סכמטי ניתן לתאר את האלגוריתם באופן הבא:

First-visit MC prediction, for estimating $V \approx v_{\pi}$

Initialize:

$\pi \leftarrow$ policy to be evaluated
 $V \leftarrow$ an arbitrary state-value function
 $Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Repeat forever:

Generate an episode using π
For each state s appearing in the episode:
 $G \leftarrow$ the return that follows the first occurrence of s
Append G to $Returns(s)$
 $V(s) \leftarrow$ average($Returns(s)$)

איור 11.5 אלגוריתם MC עבור שערק ה-function Policy. עבור כל מצב מתחילה רישמה ריקה, ולאחר מכן מיצירם המון ריצות שונות. עבור כל ריצה, עוברים על כל המצבים ובודקים מה ה- G -שללם, ומוסיפים אותו לשήמה של אותו מצב. לבסוף מחשבים את ה-state-value של כל מצב על ידי מיצוע הרשימה (=ערכי G השווים) של אותו מצב.

יש מספר יתרונות לשיטה זו: היא מספקת משערק חסר הטיה עבור ה-function, Value function, מבטיחה התוכנות אחרות מספיק איטרציות, ובמסוף ניתן לשערק באמצעותה את השגיאה. אפשר גם באוטה דרך לשערק גם את $(s, a) \in \mathcal{Q}^{\pi}$ אך זה יהיה יותר רושש ודורש יותר דוגמאות. מן הצד השני יש גם חסרונות לשיטה זו: ראשית, היא מומאייה רק ל- Episodic MDP ולא לריצות אינסופי.ites. עם בעיה זו ניתן להתמודד בקהלות כיון שעבור בעיה אינסופית ניתן לקחת ריצה סופית וליחסם את השגיאה באמצעותה. שנית, ההתקנות יכולה להיות מאוד איטית, ובמסוף השונות יחסית גובהה.

11.2.2 Temporal Difference (TD) – Bootstrapping

במוקם להסתכל על ריצה שלמה, ניתן לאחרי כל עודן את ה-state-value. מושוואת בלמן ניתן להראות שהbijוי $\mathcal{V}^{\pi}(S_{t+1}) + \gamma \mathcal{V}^{\pi}(S_{t+2})$ הינו משערק הסיטה עבור $(S_t)^{\pi}$. כלומר ניתן להשתמש בשערק עבור אונחון לא יודע את ה-state-value. Value function הביטוי $(S_{t+1})^{\pi} \mathcal{V}^{\pi}$ לא ידוע. בשביל לכך ניתן לשערק את $(S_t)^{\pi}$ באמצעותו מושערק, ניתן להחליפּ את $(S_{t+1})^{\pi}$ ב- $\mathcal{V}^{\pi}(S_{t+1})$, ולבצע את השערק באופן הבא:

$$\mathcal{V}^{\pi}(S_t) = R_{t+1} + \gamma \mathcal{V}(S_{t+1})$$

הרעיו מתחוו השימוש הזה הוא להיעזר במידע שיש לנו מ- R_t . נניח ונניח ערך כלשהוא עבור $(S_t)^{\pi} \mathcal{V}$ וניחס שזיהה גורע. אפשר מעת לשבור את הניחוש באמצעות ניחוש $\mathcal{V}(S_{t+1})$ ושים ש- R_{t+1} reward עבור אותו מצב S_t , שבעצם מספק מידע כלשהו על מצב זה. שערק זה עדיף על ניחוש מוחלט כיון שהאלמנט של הניחוש מוביל מושקל נמוך יותר עקב המכפלת ב- γ , ויש יותר מושקל $R_{t+1} - R_t$ שמספק מידע אמוי על המצב S_t . צריך לשים לב שאנו מנסים לשערק את ה-function-value מתוך הערכיהם שלה עצמה. מתחווים את כל הערכים במספרים כלשהם (למשל – וקטורי של 0), ואז ערבאים צעד אחד ומדכינים את הניחושים בעוררת התגמלים. אינטואטיבית זה נראה מעט שונה, אך מסתבר שפתרון זה הוא אחד הכלים החזקים בעקבות RL. באופן פורמלי האלגוריתם נראה:

Tabular TD(0) for estimating v_π

Input: the policy π to be evaluated
 Initialize $V(s)$ arbitrarily (e.g., $V(s) = 0$, for all $s \in S^+$)
 Repeat (for each episode):
 Initialize S
 Repeat (for each step of episode):
 $A \leftarrow$ action given by π for S
 Take action A , observe R, S'
 $V(S) \leftarrow V(S) + \alpha[R + \gamma V(S') - V(S)]$
 $S \leftarrow S'$
 until S is terminal

איור 11.6 אלגוריתם Temporal Difference (TD) עבור שערוך-h-ion-value function Policy. מוחשים ערך עבר כל מצב ואז באופן איטריבי משפרים את הניחושים בערך שערוך התלוי-ב-reward ובערך המצב הבא.

מגדירים את השגיאה של TD כהפרש שבין היחס ערך המצב לבן השערוך של:

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$$

בשונה משערוך MC, השערוך בשיטת TD הוא בעל הטיה, אך השונות קטנה יותר. בנוסף, כיוון שבכל צעד מבצעים שיפור לערך של מצב, תהליך השערוך יותר מהיר מאשר ב-MC. הרבה פעמים מוסיפים פרמטר α לשערוך (כפי שמופיע באיור 11.6):

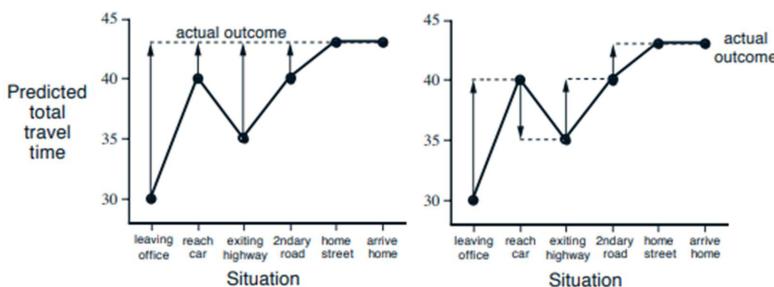
$$V(S_t) = V(S_t) + \alpha \cdot [R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$$

עבור ערכי α מתאימים, המשערוך מתקנס לתחולת האמיתית.

ניתן דוגמה שתמחיש את שיטת MC ושיטת TD ואת היחס ביניהם: נаг' יצא מהמשרד שלו וסע לbijתו, ובדרך הוא ניסה לשערוך את זמן ההגעה שלו וקיבל את הזמןים הבאים:

מצב	כמה זמן עבר	שעורך הזמן הנוכחי הכללי	שעורך הזמן הנוכחי לנסיעה
'יצאה מהמשרד'	0	30	30
הילכה למכונית תחת גשם	5	35	40
הגעה לכיבוש מהיר	20	15	35
נסעה מאחוריו משאית	30	10	40
הגעה לרחוב של הבית	40	3	43
הגעה הביתה	43	0	43

נשרטט את שני המשערוכים:



איור 11.7 שעורך MC (משמאל) וشعורך TD (מימין) ביחס לתצפית הנתונה.

כיוון שהשיטה MC מספקת ערך לאחר ריצה שלמה, אז ניקח את זמן ההגעה בפועל של הנוהג וניתן את הערך זהה לכל מצב הביניים. שיטת TD לעומת זאת מדכנת את הערך בכל מצב בהתאם למצב הבא. ניתן לראותו שהשונות בשערוך TD קטנה מזו שהתקבלה בשערוך MC.

ניתן להסתכל על שיטת TD כבעית רגסיה "динמית":

עבור כל צעד נרצה $\mathcal{V}(S_t) \approx \text{ויה שווה למשוואות בilm}$, כלומר נרצה לשער את $(S_t) \mathcal{V}$ כך שיתקיים:

$$\mathcal{V}(S_t) = \mathbb{E}_\pi[R_{t+1} + \gamma \mathcal{V}(S_{t+1}) | s_t = S_t]$$

עבור בעיה זו נוכל להגיד פונקציית מחר (Loss) מתאימה:

$$L = \frac{1}{2} (\mathbb{E}_\pi[R_{t+1} + \gamma \mathcal{V}(S_{t+1}) | s_t = S_t] - \mathcal{V}(S_t))^2$$

את הפונקציה זו ניתן למצוור עלי דגימות סטטיסטיות של $R_{t+1} + \gamma \mathcal{V}(S_{t+1})$. נשים לב לדבר חשוב – המטרה שלנו היא לשער את ההואה בנסיבות העתיד ולא היפך, כיון שהעתיד הוא בעל יותר מידע – והוא ראה reward ומצב חדש. הבדיקה זו מושפעה על איך שאנו מעריכים – וכאן שפונקציית המחיר תתרנגול – אנחנו מעריכים $\mathcal{V}(S_t)$ ותקרב לערך של $\mathcal{V}(S_t) = \mathbb{E}_\pi[R_{t+1} + \gamma \mathcal{V}(S_{t+1}) | s_t = S_t]$ ולא להיפך. בדוגמה של הנגר שמשער תמורה – הוא רוצה לעדכן את השערור שלו בהתאם למצב הבא. אם למשול ההערכה שלו בזמן t היה 30 דקות ואז הוא מגיע לפיקק וمعدכן את ההערכה ל-35 דקות, אז הוא ירצה לתקן את השערור הקודם ($\mathcal{V}(S_t)$) כך שהיא דומה לשערור הנוכחי $(R_{t+1} + \gamma \mathcal{V}(S_{t+1}))$, ולא לעדכן את השערור הנוכחי כך שהיא דומה לקודם. בכך לדאוג לכך, נתיחס לעתיד קבוע ולא נחשב עבור גרדיאנט (למרות ש- \mathcal{V} מופיע בו). באופן פורמלי נוכל לנסח זאת כך:

$$T(S_t) = \mathbb{E}_\pi[R_{t+1} + \gamma \mathcal{V}(S_{t+1}) | s_t = S_t]$$

$$L = \frac{1}{2} (T(S_t) - \mathcal{V}(S_t))^2$$

ואז הגדרינגן יהיה:

$$\frac{\partial L}{\partial \mathcal{V}} = T(S_t) - \mathcal{V}(S_t)$$

בהתאם לכך, הדגימה $R_{t+1} + \gamma \mathcal{V}(S_{t+1}) - \mathcal{V}(S_t)$ הינה שערור סטטיסטי לוגדיאנט. כיון שכל צעד תליי בצעד הבא, איז מטרת $T(S_t)$ משתנה בכל צעד (אם נמנם קיבענו אותה בכל צעד יחיד, אך היא עדין תלויה ב- $\mathcal{V}(S_t)$). במבנה זהה בעית רגסיה שהגדרנו הינה "динמית", כיון שהמטרה משתנה בכל צעד.

11.3.2 TD(λ)

נסזה לבחון את הקשר בין שתי השיטות שראינו. שערור TD מבצע דגימה של ריצה ואז מעדכן את הערך של כל מצב בהתאם למצב הבא בלבד:

$$\mathcal{V}(S_t) = \mathcal{V}(S_t) + \alpha \cdot [R_{t+1} + \gamma \mathcal{V}(S_{t+1}) - \mathcal{V}(S_t)]$$

בגלל ההתייחסות למצב אחד בכל פעם השונות של המשערר נמכה, אך יש הטיה.

שיטת MC לעומת זאת דוגמת ריצה וمعدכנת את הערך של כל מצב בהתאם לכל המצבים שבאים לאחר מכן:

$$G_t = R_{t+1} + \gamma R_{t+2} + \dots + \gamma^{T-1} R_T = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1}$$

$$\mathcal{V}(s_t) = \mathcal{V}(s_t) + \frac{1}{N(S_t)} (G_t - \mathcal{V}(s_t))$$

משערר זה הינו חסר הטיה, אך עם זאת השונות שלו גבוהה.

נראה כיצד ניתן לחבר בין שני המשעררים ולמצואו שיטה שתהיה אופטימלית מבחינת היחס שבין ההטיה לשונות. ניתן להכליל את שני המשעררים לנוסחה כללית באופן הבא:

$$\mathcal{V}(s_t) = \mathcal{V}(s_t) + \alpha (G_t^{(n)} - \mathcal{V}(s_t))$$

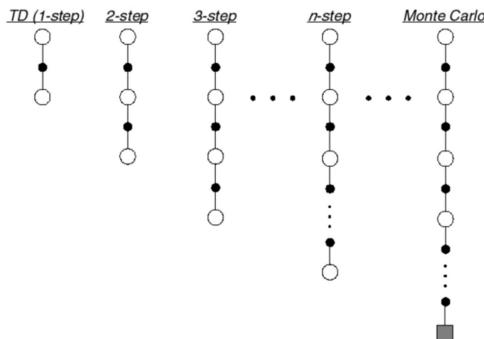
כאשר $G_t^{(n)}$ הוא הסכום של n rewards הבאים:

$$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n \mathcal{V}(s_{t+n}) = \sum_{k=0}^{n-t-1} \gamma^k R_{t+k+1}$$

כעת נוכל להגיד:

$$TD(n) \rightarrow \mathcal{V}(s_t) = \mathcal{V}(s_t) + \alpha (G_t^{(n)} - \mathcal{V}(s_t))$$

ולפי סימן זה נוכל לשים לב שמערך MC הוא למעשה $\infty \rightarrow n$ TD ואילו שמערך TD שווה -1 .



איור 11.8 משורך MC הינו המקהה הפרטיאי בו $n = 1$. $TD(n)$ משורך MC הינו המקהה הפרטיאי בו $n > 1$

אם ניקח $n < \infty$ נקבל משורך "מוצע" בין MC לבין TD. ניתן להציג גרסה יותר טובת למשערך זה תחת ההנחה שככל שמאורעут סמכים אחד לשני קר יש להם יותר השפעה. בדומה ל- λ discount factor שMOVED את הרשيعة של תגמול כל שווא יותר רוחק מהמצב הנוכחי, קר גם כאן ניתן לכל מארע עתידי משלך הולך וקטן. נdag שכל המשקלים יסתכם ל-1 ונקבל את השערך הבא, הידוע גם בכינוי $(\lambda)TD$:

$$G_t^\lambda = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

$$\mathcal{V}(s_t) = \mathcal{V}(s_t) + \alpha (G_t^\lambda - \mathcal{V}(s_t))$$

שערך זה הוא בעל שונות גודלה יותר מאשר TD, כיון שהוא מתחשב ביוטר צעדים, אך ההטייה שלו קטנה יותר. כאמור, הוא מנסה למצוע בין שני המשערכים שראינו ולאין בין השונות להטייה.

נסכם בקצירה את מה שראינו בפרק זה. התיחסנו למקרים בהם המודל אינו ידוע לנו ורוצים לשערך אותו, והציגנו שלוש גישות המנסות לשערך את המודל בעזרת דגימות סטטיסטיות: MC, TD (שהזה מקהה פרטיאי של n) וגישה המנסה לשלב בין השתיים – $(\lambda)TD$.

נסים נעיר ששערך האסטרטגיה כשלעצמה אינם מספיק, כיון שהשערך אמן מספק אסטרטגיה מסוימת עבור הבעה, אך היא אינה בהכרח אופטימלית. בפרק הקודם רأינו כיצד בהינתן המודל ניתן לשפר אותו ולמצוא את האסטרטגיה האופטימלית. ככלנו עושים זאת כיוון שדענו את המודל במלואה, מה שאפשר לבצע בכל צעד מהלך חסדי ובקר להתבסס לבסוף לאסטרטגיה האופטימלית. במרקם בהם אנו רק משערכים את האסטרטגיה ללא דיעת המודל, אין לנו בהכרח מידע מספיק טוב עבור כל המצבים. מצבים ופעולות שלא נסעו אך בפעמים נדרות, אין לנו בהכרח מידע מספיק טוב עבור כל המצבים. שיטות שלא נסעו אך בהכרח יכול להביא את האסטרטגיה להיות אופטימלית.

11.3 Model Free Control

בפרק הקודם רأינו כיצד ניתן לשערך את המודל באמצעות דגימות סטטיסטיות. שיטות השערך אפשרו לנו לקבל מידע על המודל, אך הן אינן התייחסו לשאלת האם הוא אופטימלי. בפרק הראשון רأינו כיצד ניתן לקחת מודל או

אסטרטגיה ולהביאו אותם לאופטימליות, אך אי אפשר להשתמש בשיטה זו עבור מודל משוער. הסיבה לכך נעוצה שמודל זה לא בכחלה ראה את כל הזוגות האפשריים של הפעולות והאפשרויות, ומילאiao הוא לא יכול לשלוף את הבחירה שלו על סמך אסטרטגיה גיידית. ניקח לדוגמה מקרה בו עבור מצב מסוים האסטרטגיה הינה דטרמיניסטית והפעולה שנבחרת הינה תמיד ללכת לעלה. במקרה זה אין לנו שום מידע על יתר הפעולות האפשריות במצב זה וכן מודל שלנו לא שלם, ומילאiao לא יוכל להשתמש ב-**Policy improvement** המבוסס על כך שיש לנו מידע על כל המצבים והפעולות.

בדי' להתמודד עם בעיה זו נהיה חייבים לדאוג לכך שבקר בכל המצבים. כמובן שוכן למקוטם בגישה פשטיינית של אסטרטגיה אקראיית, שאחרי מספיק גדול של פעולות קרוב לוודאי שנבחר בכל המצבים. אסטרטגיה זו אמןנו טוביה לבדיקה מצבים ופעולות חדשים, אך כמובן שהיא רוחקה מיליה אופטימלית, שכן נרצה להשתמש באסטרטגיה שמצוד אחד מונחה להיות אופטימלית ומצד שני יש בה ממד של אקריאות המביא לכך שבקר גם במצבים שלא יינו מגייעים אליהם לפי האסטרטגיה הנוכחית. בחירה פעולות בהתאם לאסטרטגיה נקראת **נקראת Exploration** ואילו בחירה של מצבים חדשים נקראת **Exploration**, והמטרה שלהם תהיה לאזן בין השנים תחת הדרישות הבאות:

1. ביצורו בכל הזוגות של המצבים והפעולות אינסוף פעמים.

2. ככל שמספר הצעדים גדל, כך האסטרטגיה מתכנסת לאסטרטגיה הגידית:

$$\lim_{k \rightarrow \infty} \pi_k(a|s) = 1 \left[a = \arg \max_{a'} Q(s, a') \right]$$

מצד אחד נרצה לבקר בכל המצבים ומצד שני נרצה שכל שנעשה יותר צעדים ככה נשפר את האסטרטגיה שלנו. אסטרטגיה המקיים דרישות אלה נקראת **Greedy in the Limit of Infinite Exploration (GLIE)**, וניתן להוכיח שקיים דרישות אלה מביא את האסטרטגיה להיוות אופטימלית. דוגמה לאסטרטגיה פשוטה העונה על הדרישות הינה $\epsilon - \text{greedy}$ – בחירה של האסטרטגיה האופטימלית בהתקבשות $(1 - \epsilon)$ ובוחרת מצב אחר בהסתברות ϵ . לעומת ϵ שהולך וקטן עד -0 בקצב שיאנו מהר מדי, אסטרטגיה זו הינה GLIE.

על הראיינו כיצד בניתוח מודל ניתן לשפר את האסטרטגיה (Policy improvement) על ידי כך שבחורנו באופן גרייד' פעולות. נתן נרצה להראות שבאופן דומה מתקיים אותו רעיון גם עבור $\epsilon - \text{greedy}$ w.r.t Q^{π_i} , כלומר שאמם השתמשנו בשיטה זו ותגננו ל-Value function, אז ניתן לבצע עבור את הערך $V^{\pi_{i+1}}$ על ידי אסטרטגיה ϵ -גראידית. אסטרטגיה זו בוחרת באופן גרייד' את הפעולה האופטימלית לפי האסטרטגיה הנוכחית, אך נותנת לכל יתר הפעולות הסתברות הגדולה או שווה להסתברות שהיא להפעלה נושא ב- $Q^{\pi_{i+1}}$ שימוש באסטרטגיה ϵ -גראידית. נסוח את המשפט באופן פורמלי:

If Policy π_i has $\forall s, a: \pi_i(a|s) \geq \frac{\epsilon}{|A|}$ and π_{i+1} is $\epsilon - \text{greedy}$ w.r.t Q^{π_i} , then $V^{\pi_{i+1}} \geq V^{\pi_i}$

הוכחה: נסתכל על המקרה בו נוקטים צעד אחד גרייד' לפי π_{i+1} ואז ממשיכים לפי האסטרטגיה הקודמת π_i :

$$\sum_a \pi_{i+1}(a|s) Q^{\pi_i}(s, a) = \sum_a \frac{\epsilon}{|A|} Q^{\pi_i}(s, a) + (1 - \epsilon) \max_a Q^{\pi_i}(s, a)$$

נרשום את $(\epsilon - 1)$ בצד ההפוך: במקומות $\sum_a \pi_i(a|s) = \frac{\epsilon}{|A|}$, ובמקום 1 נרשום $\sum_a \pi_{i+1}(a|s)$ (הסכום אכן שווה $1 - 1 = 0$). סוכמים את כל האפשרויות עבור התפלגות נתונה. נקבל:

$$= \sum_a \frac{\epsilon}{|A|} Q^{\pi_i}(s, a) + \max_a Q^{\pi_i}(s, a) \sum_a \left(\pi_i(a|s) + \frac{\epsilon}{|A|} \right)$$

כעת נחליף את הביטוי $\sum_a \pi_i(a|s)$ באמצעות $\sum_a \pi_{i+1}(a|s) = \frac{\epsilon}{|A|}$. ניקח שמתקיים $Q^{\pi_i}(s, a) \leq \max_a Q^{\pi_i}(s, a)$ ב- $Q^{\pi_i}(s, a)$ ונקבל:

$$\geq \sum_a \frac{\epsilon}{|A|} Q^{\pi_i}(s, a) + Q^{\pi_i}(s, a) \sum_a \left(\pi_i(a|s) - \frac{\epsilon}{|A|} \right)$$

כעת יש שני איברים זהים שמצטמצמים, ונשאר עם:

$$= \sum_a \pi_i(a|s) Q^{\pi_i}(s, a)$$

ובסך הכל קיבלנו:

$$\sum_a \pi_{i+1}(a|s) Q^{\pi_i}(s, a) \geq \sum_a \pi_i(a|s) Q^{\pi_i}(s, a)$$

הביטוי שהתקבל מסמל את ה-value function π כאשר עוקבים אחר האסטרטגיה π . יצא מכך שאמם עושים צעד אחד לפי האסטרטגיה π ואז ממשיכים לפי π , זה בהכרח יותר טוב מאשר גם את הצעד הראשון לעשות לפי π .icut בדומה להוכחה שהראינו לעיל, ניתן להוכיח שמתוקים $(s) \leq V^{\pi_{i+1}}(s) \leq V^{\pi_i}(s)$ וביצוע השיפור שוב ושוב יביא את האסטרטגיה להתכנס לזו האופטימלית. הבניה בשיטה זו היא חוכר העולות שבה, כיוון שהוא דורשת המNON דוגמאות והמון איטרציות. עקב קר שיטה זו לא פרקטית, ובמקרה הבא ציג כתעת שיטות אחרות המאפשרות לשפר את האסטרטגיה עברו מודל משוער. פורמלית, שיטות אלה ננסחות תחת קטגוריה הנקראת Model Free Control – שליטה (ושיפור) אסטרטגיה שאינה מtabסת על מודל ידוע מראש.

11.3.1 SARSA – On-Policy TD control

בפרק הקודם רأינו כיצד ניתן באמצעות שערון TD לשפר את ה-value function, כאשר העדכון בכל צעד מתבצע באופן הבא:

$$V(S_t) = V(S_t) + \alpha \cdot [R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$$

כתעת אנו ממעוניינים באסטרטגייה ולא רק ב-value function, לכן בזמנים לעדכן את (S_t, A_t) ננדכן את פונקציית Q - $Q(S_t, A_t)$ state-action Q . באופן הזה נקבל טבלה בגודל $|S| \times |A|$: המכילה מידע כל הזוגות האפשריים של (S, A) , ובעבור כל זוג יש ערך מסוים (טבלה זו נקראת Q -table). בהתחלה הערכים בטבלה לא 'שקרו' או את הערכים האמיתיים, אך עם התקדמות הלמידה הطلبת תשפר ועלי אף תמכנו לאופטימליות. העדכון מתבצע באופן הבא:

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha \cdot [R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

העדכון בכל צעד מתבצע על סמך הממצבים והפעולות של שתי יחידות זמן: $\{S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1}\}$ ולכן האלגוריתם נקרא SARSA. לואוריתם זה הינו On-Policy Learning. לעומת זאת, העדכון בכל צעד נעשה על סמך מידע המופיע מהארסטרטגיה הידועה באותו זמן: בורותים לבצע פעולה A_t במצב S_t בmäßig $Q(S_t, A_t)$ בהתאם לאלגוריתם, ואז מעדכנים אותה על סמך התגמול R_{t+1} שהתקבל בעקבות הפעולה A_t . באופן סכמתי איטרציה אחת של האלגוריתם מתוארת באופן הבא:

```

Initialize  $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$ , arbitrarily, and  $Q(\text{terminal-state}, \cdot) = 0$ 
Repeat (for each episode):
    Initialize  $S$ 
    Choose  $A$  from  $S$  using policy derived from  $Q$  (e.g.,  $\epsilon$ -greedy)
    Repeat (for each step of episode):
        Take action  $A$ , observe  $R, S'$ 
        Choose  $A'$  from  $S'$  using policy derived from  $Q$  (e.g.,  $\epsilon$ -greedy)
         $Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma Q(S', A') - Q(S, A)]$ 
         $S \leftarrow S'; A \leftarrow A'$ ;
    until  $S$  is terminal

```

איור 11.7 אלגוריתם SARSA. שערון on-policy של $Q(s, a)$ בעזרת דוגמאות מאסטרטגיה ϵ -גרידית ביחס ל- Q המקורי.

בכל מצב הפעולה שנבחרת הינה ϵ -גרידית ביחס ל- $Q(a|s)$ הנוכחי. כלומר: אם במצב S_t יש לנוקוט לפי האסטרטגיה את המצב $A_t = \bar{A}$, אז האסטרטגיה ϵ - ϵ -גרידית ביחס לכך:

$$A_t = \begin{cases} \bar{A} w.p & 1 - \epsilon \\ A \neq \bar{A} & w.p \epsilon \end{cases}$$

ניתן להוכיח שאלגוריתם SARSA מביא את האסטרטגיה לאופטימליות תחת שני תנאים:
א. שהאסטרטגיה תהיה GLIE.

ב. שיתקיים תנאי Robbins-Monroe עבור α (תנאי זה דואג לכך שנגיע בהכרח לכל הממצבים ומצד שני $0 \rightarrow (\alpha_i)$:

$$\sum_{i=0}^{\infty} \alpha_i = \infty, \sum_{i=0}^{\infty} \alpha_i^2 < \infty$$

לגיישה זו, הפעלת בגישה *on-policy learning*, יש מספר חסרונות:

1. המטרה היא ללמד את האסטרטגיה האופטימלית אבל בפועל *exploration* הוא ביחס לאסטרטגיה הנתונה בכל מצב.
2. לא ניתן להשתמש בצעדים ידניים, כיון שהם מתייחסים לאסטרטגיה שכבר לא רלוונטית.
3. לא ניתן להשתמש במידע שmag'ן מבחן.

11.3.2 *Q*-Learning

ניתן להפוך את אלגוריתם SARSA להיות *off-policy*, והאלגוריתם המתkeletal, שנקרא *:SARSA* או אחד האלגוריתמים השימושיים בתחום של RL. נבונים בפונקציית העדכון של

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha \cdot [R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

האלמנט שטלוי באסטרטגיה הינו $Q(S_{t+1}, A_{t+1})$, כיון שבנו אותו בפועלה A_{t+1} בהתאם לאסטרטגיה. במקום לבחור בפועלה זו, ניתן להיות גרייד ולחתת את הפעולה בעלת הערך היכי גדול בצד הקרוב, ועל פיה לעדכן את ה-*:Q-value*

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha \cdot [R_{t+1} + \gamma \max_A Q(S_{t+1}, A_t) - Q(S_t, A_t)]$$

cut האסטרטגיה אינה משפיעה על פונקציית העדכון, ומילא היא off-policy

```

Initialize  $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$ , arbitrarily, and  $Q(\text{terminal-state}, \cdot) = 0$ 
Repeat (for each episode):
    Initialize  $S$ 
    Repeat (for each step of episode):
        Choose  $A$  from  $S$  using policy derived from  $Q$  (e.g.,  $\epsilon$ -greedy)
        Take action  $A$ , observe  $R, S'$ 
         $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$ 
         $S \leftarrow S'$ ;
    until  $S$  is terminal

```

איור 11.8 אלגוריתם *Q*-Learning. שערוך $Q(s, a)$ של *off-policy* באמצעות דוגמאות מאסטרטגיה ϵ -גראדיית ועדכן ה-*Q-value* בהתאם לפועלה בעלת הערך היכי גדול בצד הקרוב.

האסטרטגיה לא משפיעה על עדכון ה-*Q-value*, אך כן יש לה השפעה על **כמויות** הפעמים שנברך בכל מצב. בኒיגוד לאלגוריתם SARSA בו דרשנו שהאסטרטגיה תהיה GLIE, באלגוריתם *Q*-Learning Q הדרישה הינה רק שנברך אינטוף פעמים בכל מצב. הבדל זה הוא משמעותי, כיון ש-GLIE דורש שהאסטרטגיה תתכנס לאסטרטגיה הגראדיית (כלומר, הדרישה היא ϵ -ילך-0). דרישת ה- ϵ -מוניטין מחייבת ש- Q יהיה שצעדים גריידיים מביים מעט מיותר חדש. הסרת הדרישה על ה- ϵ מאפשרת exploration בצורה יותר חופשית, ולהלמידה נועשית בצורה הרבה יותר מהירה. עם זאת, יותר מדי exploration זה גם לא טוב, כיון שמקרים בהרבה מצבים לא רלוונטיים מספר רב של פעמים.

נתבונן על האלגוריתמים שראינו עד כה ונשווינה בין הפתרונות שניים מבוססים על ידיעת המודל לבני פתרונות משוערכים:

Full Backup (Dynamic Programming)	Sample Backup (TD)
Iterative Policy Evaluation: $V(S) = \mathbb{E}[R + \gamma \cdot V(S') S]$	TD Learning: $V(S) = V(S) + \alpha \cdot [R_{t+1} + \gamma V(S') - V(S)]$
Q-Policy Iteration: $Q(S, A) = \mathbb{E}[R + \gamma \cdot Q(S', A') S, A]$	SARSA: $Q(S, A) = Q(S, A) + \alpha \cdot [R + \gamma Q(S', A') - Q(S, A)]$
Q-Value Iteration:	Q-Learning:

$$\mathcal{Q}(S, A) = \mathbb{E} \left[\mathcal{R} + \gamma \cdot \max_A \mathcal{Q}(S, A) | S, A \right] \quad \mathcal{Q}(S, A) = \mathcal{Q}(S, A) + \alpha \cdot \left[\mathcal{R} + \gamma \max_A \mathcal{Q}(S', A) - \mathcal{Q}(S, A) \right]$$

הפתרונות המשוערים מצלחים להתמודד עם בעיות ביןויות, אך הצורך בדges מושתים סיבוט: א. האלגוריתמים נדרשים להמן דגימות בשבייל להצלחה. ב. חסרון נוסף בגין שאלות שאינן model-based exploration עבר בעיות של העולם האמיתי, לא תמיד אפשר לראות דוגמאות שליליות כדי למדוד שהן לא טובות. רכב אוטומטי למשל, אם רוצה שהוא יימדך לא לנוסע ברמות אדום, אין אפשר לאמן אותו על ידי זה שהמיתון יידחופש לתביעה exploration וכך הוא יוכל למסובים בהם הוא ייסע באורוור ויקבל ערך תגמול שלילי. לעומת זאת, בעיות הכרוכות בסיטואציה הן יותר מתאימות לאלגוריתמים שהציגו המבוססים על דגימות ושורר, כיוון שיתן בקרה יחסית זוליה לבצע המזון דגימות, ובנוסף אין בהן בעיות ביחסו exploration-horizon.

מלבד בעיית הדges והשערון, האתגר העיקרי של הכלכלה של המודלים המלדים. התוצאה של SARSA – \mathcal{Q} – הינה כאמור טבלה של-value – \mathcal{Q} , ובטבלה זו אין שום קשר בין הערכים השוניים בטבלה. כל איבר בטבלה עומד בפני עצמו, ואילו אפשר ללמד ממנה על שאר האיברים. אם למשל הענו למספר החדש שעוד לא ריאנו אך ראיינו הרבה ממצבים דומים לו, לא יוכל ללמד מהם שום דבר לגבי המצב החדש. אתגר זה משולץ גם על גודל הביעות אותן ניתן לפרט – אם אין אפשרות להכליל ממצב אחד לנצח אחר, מילא זה מגביל מאוד את גודל הבעה אותה ניתן להתמודד בעדרת אלגוריתמים אלו. במקרים בהם מרכיב הממצבים הוא צפוף, אז מרכיב הממצבים הוא אינסופי ואילו בכלל לא ניתן להשתמש בගישות אלו.

11.3.3 Function Approximation

כאמור, אלגוריתמים שמנוטים לשערר את-table – \mathcal{Q} אינם ישנים בעיות גדולות בעקבות חוסר היכולת שליהם להכליל ממצב אחד למצב אחר. גם אם יש בידינו אפשרות לשמור טבלאות ענקיות של-value – \mathcal{Q} , לא יוכל לשפר את הטבלה ולעתק בה ערכים אוטומטיים, כיון שלא ניתן להשליך ממצבים בהם ביקרתו על ממצבים חדשים. בכך להתמודד עם בעיה זו, נרצה להחליף את-table – \mathcal{Q} בפונקציה שמחזירה ערך עבור כל זוג של state-action. במקום לחשב טבלה ענקית $|A| \times |S|$: \mathcal{Q} , נרצה למצוא פונקציה עם סט פרמטרים θ כך שיתקיים:

$$\mathcal{Q}(S, A) \approx \mathcal{V}_\theta(S)$$

היתרון של שימוש בפונקציה שמנסה לשערר את הערכים הינם כפול: א. לא צריך לשמר טבלה בגודל של מרכיבים. ב. כן ניתן ללמידה על עלייהם יעד על ממצבים חדשים. בכך למצוין פרמטרים שישרעו את המודל בכוונה אינטואיטיבית יש לפטור בעית אוטומטייה, כפי שנגיד בהמשך, אך הלמידה יכולה להיות מאוד לא יציבה. ראשית, כפי שראינו ב-TD, המטרה אויה רצף לשערר (S_t, \mathcal{Q}) או (S_t, \mathcal{V}) בפונקציה שמחזירה ערך בכל צעדי. בנוסף, ביגוד לאלגוריתמים הקודמים,icut יש קשר בין הממצבים, ולכן אם אנחנו מנסים ערך מסוים, זה גם ישפיע על ערכים אחרים, מה שמקשה מאוד על יציבות הלמידה.

הדוגמה הפשוטה ביותר לפונקציה כזו הינה מודול לינארי מהצורה:

$$\mathcal{Q}(S, A) = w^T \phi(S, A)$$

מודל זה מניח שיש סט של פרמטרים המקיימים קשר לינארי בין-state-action ומפת פיצרים כלשהיא $A(S, A)$ לבין הערך שלהם $\mathcal{Q}(S, A)$. באופן הפשטני ניתן, בשוביל למצאו את הפונקציה הרצויה, ונרצה למצויר כמה שיותר את המרחק שבין (S, \mathcal{V}) לבין (S, \mathcal{Q}) , ולשם כך נבנה את פונקציית המטרה הבאה:

$$L(\theta) = \frac{1}{2} \mathbb{E}_\theta \left[(\mathcal{V}^\pi(S) - \mathcal{V}_\theta(S))^2 \right]$$

כמובן שחישוב זה אינו אפשרי משום שאין לנו יודעים מה הוא (S, \mathcal{V}^π) – זה בדיקת הביטוי אותנו רוצים לשערר! בנוסף, חישוב התוצאות יכול להיות מאד מסובך, בטח במקרה בו אנו לא יודעים את כל הערכים האינטואיטיביים של (S, \mathcal{V}^π) . בשוביל להתגבר על בעיות אלו נשים לב כיצד ניתן למצאו את הערך $\mathcal{V}_\theta(S)$ – אם נזוזר את פונקציית המטרה וביצע צעד בכיוון הגרדיינט, נקטין את ערכיה, וכך נרצה להיות מסוגלים לחשב את $\Delta\theta$. בפועל זאת בדומה בעדרת דגימות סטטיסטיות, כפי שכבר ריאנו בפרקם קודמים, כמובן נרצה לדגום ממצבים מהסטרטגיה בשוביל לחשב את הביטוי הבא:

$$\Delta\theta = (\mathcal{V}^\pi(S_t) - \mathcal{V}_\theta(S_t)) \nabla_\theta \mathcal{V}_\theta(S_t)$$

עם ביטוי זה עדין לא ניתן להתקדם כיון ש- π^{θ} לא ידוע, אך גם אותו ניתן לשער, למשל בעזרת MC או TD. בעזרת שיטת MC ניתן להריץ episode שלם, לחשב את התגמול המצביע ולשיט אותו במקום $\pi^{\theta}(S)$, ובעזרת TD ניתן להריץ צעד אחד ולהחליף את π^{θ} בתגמול המידי והשערו של המצב הבא:

$$MC: \Delta\theta = (G_t - \mathcal{V}_{\theta}(S_t))\nabla_{\theta}\mathcal{V}_{\theta}(S_t)$$

$$TD: (R_{t+1} + \gamma\mathcal{V}_{\theta}(S_{t+1}) - \mathcal{V}_{\theta}(S_t))\nabla_{\theta}\mathcal{V}_{\theta}(S_t)$$

יש לשים לב שבביטוי האחרון יש שני איברים שתלויים ב- θ – $\mathcal{V}_{\theta}(S_{t+1})$ ו- $\nabla_{\theta}\mathcal{V}_{\theta}(S_t)$, אך נגזר רק את הביטוי הראשון כיון שנרצה לשנות אותו קר שיתקרב לביטוי השני ולא להיפר. עם זאת, ביגוד למה שראינו לעיל בדוגמא TD, שם שניי של $(S_t) \mathcal{V}$ לא השפיע על $(S_{t+1}) \mathcal{V}$ כיון שהערכים בטבלה הוי בלתי תלויים אחד בשני, כאן כן יש השפעה בין שני ערכים אלו, מה שמקשה על היציבות של הלמידה.

בסוף דבר, אנו יכולים לחשב את הביטוי $\Delta\theta$ ובעזרתו לנסוט לאפותם את הפרמטרים באיזה שיטות אופטימיזציה סטנדרטיות (למשל – stochastic gradient descent).

בדומה לשערוך שהצענו עבור MC-TD, ניתן גם לבצע control ולשערך את ה- \mathcal{Q} – table

$$SARSA: \Delta\theta = (R_{t+1} + \gamma\mathcal{Q}_{\theta}(S_{t+1}, A_{t+1}) - \mathcal{Q}_{\theta}(S_t, A_t))\nabla_{\theta}\mathcal{Q}_{\theta}(S_t, A_t)$$

$$\mathcal{Q}-Learning: \Delta\theta = \left(R_{t+1} + \gamma \max_A \mathcal{Q}_{\theta}(S_{t+1}, A_t) - \mathcal{Q}_{\theta}(S_t, A_t) \right) \nabla_{\theta}\mathcal{Q}_{\theta}(S_t, A_t)$$

וגם כאן, בגזירה נרצה להתייחס לכך $\mathcal{Q}_{\theta}(S_{t+1}, A_{t+1})$ – קובעים ולגזר רק את $(S_t, A_t) \mathcal{Q}_{\theta}$, כיון שנרצה לשנות את $(S_t, A_t) \mathcal{Q}_{\theta}$ קר שיתקרב בערכו לשאר הביטויים המבוססים על מידע נוסף.

יש כמה אתגרים שיכולים להקשוט מואוד על שיטות אלה להבא את המודל להתקנות:

א. ראשית, הדרישה שאומרת לנו רצים לשנות את $(S_t, A_t) \mathcal{Q}_{\theta}$ ביחס למידע העתידי הינה בעייתית, כיון שכעת כל הביטויים תלויים באותו פרמטר θ .

ב. באלגוריתם שהוא off-policy אנו דוגמים מאסטרטגיה ואך פועלם לפי הדגימות האלו, אך אנו לא מנסים להביא לאופטימום דואקן את האסטרטגיה זו (ומילא היא לא בהכרח האופטימלית עבורו). כאשר ניסינו ללמידה tabular Q-table, זה היה יכול לגרום לכך שכיחות הממצבים שנציגים אינה תואמת את השכיחות שלהם לפי האסטרטגיה האמיתית, מה שיכל להשפיע על קצב הלמידה, אך זה לא יגרום לשגיאות באסטרטגייה הנלמדת. כתע כשר יש קשר בין הממצבים על ידי הפרמטר θ , אז דוגמה לא לפי השכיחות האמיתית יכולה להטעות את הלמידה לכיוונים שגויים.

אתגרים אלו מופיעים על יכולת המודל להתקנס לאסטרטגיה האופטימלית. בעוד שבעור יכולנו להוכיח התקנסות, עבור functional-approximation זה כלל לא מבוטה, כפי שמצוג בטבלה הבאה:

Algorithm	Tabular	Linear	Non-Linear
MC	Y	Y	Y
SARSA	Y	Y	N
Q-Learning	Y	N	N

אחד השיטות פורצות בדרך התchromה הייתה שימוש ברשתות נירונים בתרח ה-*functional-approximation*, ובשם הנפוץ שלה – Deep Learning. השימוש ברשת נירונים בפני עצמה לא הייתה מספקה כיון שכאמור הלמידה היא מאוד לא יציבה ויש להכניס מנגנונים נוספים שייעזרו למודל להשתפר. נציין שתי טכניקות שהוטמעו בתהילך הלמידה:

1. כאשר דוגמים כל מיני ממצבים, יש בינהם קובליצה יחסית גבוהה, מה שמנע מהשנות של הגרד'אנט לקטן. כדי להתגבר על כך, במקומות לעדכן את \mathcal{Q} בכל שלב, שומרים את הריבועיות $(S_t, A_t, R_{t+1}, S_{t+1})$ של N הממצבים האחרונים (למשל: $N = 1,000$), ואך מעדכנים לפ' k ממצבים (למשל: $k = 10$) אקראים מתוך N הממצבים הקודמים שיש לנו בזיכרון (לאחר שהזיכרנו מותמלא, אך כל מצב חדש שמגיע דורש את המצב היכי ישן בזיכרון). בקרה הזו

השונות תקען כי הדוגמאות שנעדכו לפיהן יהיו בעלות קורלציה ונוכחה הרבה יותר. המחיר של טכניקה זו הוא השימוש בהרבה זיכרון, אבל ש לה השפעה משמעותית על הביצועים.

2. הזכרנו לעיל שגם רצים לשנות את $(S_t, A_t) \xrightarrow{\theta} Q$ ביחס למידע העתידי, אך זה יכול להיות בעיתוי, כיוון שכעת כל הביטויים תלויים באותו פרמטר θ . בימים אחרים – אנו רצים לקרב את האסטרטגיה למטרה מסוימת, אך שניתן תלויות באותו פרמטר. במקרה של ייצב את הרוגטיה, ניתן לשנות את המטרה באופן הרבה פחות תדר. באופן פורמלי, נשמר סט פרמטרים θ ונשנה אותו כל מספר קבוע של צעדים לסט הפרמטרים החדש, ככלומר – כל k צעדים נעדכן את θ בהתאם ל- R_t . יחד עם הפרמטרים החדשים, נשנה מעט את הביטוי אותן נרצה להביא לאופטימום:

$$\mathbb{E}_{S_t, R_{t+1}, A_t, S_{t+1}} [R_{t+1} + \gamma \max_A Q_\theta(S_{t+1}, A_t) - Q_\theta(S_t, A_t)]$$

11.3.4 Policy-Based RL

יש כמה מגבלות וחסכנות בשיטות שראינו עד כה. שיטות אלו מתמקדות בلمידת-table- Q , בין אם על ידי עדכון ישיר של ערכי הטליה ובין אם על ידי למידת הפונקציה $(S, A) \xrightarrow{\theta} Q$ התלויה בפרמטר θ . ראיינו שבבעיות גדולות הייצוג והלמידה יכולים להיות קשים. אටזר זה מטעצם אף נהיה בלתי אפשרי עבר בעיות רציפות, בהן מרחיב המבצעים הוא כביכול אינסופי (למשל – תנעה של רובוט). חסוך סופף שיש בגשות שראים נוץ במרקחה של הלמידה – והוא מנוסים ללמידה את $(S, A) \xrightarrow{\theta} Q$ אך בפועל מה שמעוניין אותנו זה $\max_A Q_\theta(S, A)$, מה שיכלול להבא לΖבדוד מושגים עבור למידה של דברים לא הכרחיים. למשל – אם אנחנו צריכים להחליט בין להשיקע כסף בשוק ההון לבין השקעה בתרמיית פירמידה, אז אנחנו רק צריכים להחליט איזו אופציה עזיפה לנו, אך אנו לא נדרשים לדעת מה בדיקת כל אופציה. ההחלה בין האופציות היא פשוטה, ואילו ללמוד מה ואיך להשיקע בשוק ההון זה דבר מאד מוכבל. لكن במקרה ללמידה את כל המערכת, נוכל פשוט ללמוד מה הפעולות הנדרשות עבורנו, וגישה ישירה זו יכולה לחסוך במשאבים.

כאמור, במקרים ללמידה את Q , ננסה ללמידה באופן ישיר את האסטרטגיה האופטימלית – $(s|a)_\theta$. באופן מעט יותר פורמלי נגידור את פונקציית המטרה:

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

מציאת האסטרטגיה האופטימלית שקופה למציאת המקסימום של פונקציית המטרה $J(\theta)$.

נרצה להשתמש באסטרטגיה סטוכסית מכמה סיבות. ראשית, הרבה פעמים יותר קל למציאו אופטימום עבור בעיה רציפה מאשר בעיה דיסקרטית, ואסטרטגיה סטוכסית יכולה להפוך בעיה דיסקרטית לרציפה. שנייה, היו מקרים בהם נתענין דוקא באסטרטגיה הסטוכסית ולא בזו הדטרמיניסטית. בנוסף, הסטוכסיות מאפשרת exploration, וכי שראינו זה מופיע הCarthy' בשבל להגיע לכל המבצעים. במבט יותר, אסטרטגיה סטוכסית היא הכללה של אסטרטגיה דטרמיניסטית – אם כל הסתברויות של המודול הן 0 או 1. בתום זהה, השימוש באסטרטגיה סטוכסית לא ממעט מהיכולת של אסטרטגיה דטרמיניסטית אלא הוא רק מقلיל אותה.

באופן הכל פשוט, עבור בעיה בה מרחב הפעולות הינו דיסקרטי, ניתן להפוך אסטרטגיה שמספקת ערך לכל פעולה $\phi(a; x; \theta)$ לההפלגות על ידי SoftMax:

$$\pi_\theta(a|s) = \frac{\exp(\phi(a; x; \theta))}{\sum_{a'} \exp(\phi(a'; x; \theta))}$$

אם מרחב הפעולות הוא רציף, ניתן להשתמש בגאון:

$$\pi_\theta(a|s) \sim \mathcal{N}(\mu(x; \theta), \Sigma(x; \theta))$$

ואם רצים לאפשר יותר חופש פעולה, ניתן להשתמש ב-Gaussian mixture model. הפונקציה שבאמצעותה רצים ל"יצא את האסטרטגיה" יכולה להיות כרצוננו – פונקציה לינארית, רשת ניירונים וכו'.

רוב האלגוריתמים שעושים אופטימיזציה $-L(\theta)$ מבוססי גרדיאנט, אך ישם גם אלגוריתמים אחרים, כמו למשל אלגוריתמים גנטיים, אך עליהם לא דобра. הדרך הסטנדרטית לבצע אופטימיזציה היא להשתמש ב-gradient descent/ascent ובפיתוחים שלו (כמו למשל $\text{stochastic gradient descent/ascent}$), והמוקד שלנו יהיה בשאלת

כיצד לשערך את הגרדיינט ופחות נטעק בשיטות האופטימיזציה השונות. האלגוריתם הבסיסי מבוסס על ערךון מהצורה הבאה:

$$\theta_{t+1} = \theta_t + \alpha \nabla f(\theta)$$

נניח ויש לנו MDP episodic, ומהאסטרטגיה הקיימת נוכל לדוגם N מסלולים $\pi_\theta(\tau_0, \dots, \tau_1, \tau)$, כאשר כל מסלול הינו $G_i = \sum_{t=0}^{T(i)} \gamma^t R_{t+1}^{(i)}$. שערוך מונטה-קרלו הרgel ישערך את המטרה באופן הבא:

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}_{t+1} \right] \approx \frac{1}{N} \sum_i G_i$$

הבעיה היא שהאפקט של הפרמטר θ על המסלול G_i היא לא ישירה – אנחנו דוגמים מסלול מאסטרטגיה π_θ התלויה בפרמטר θ , אך מחשבים את התגמול המסלול וממנו מרכיבים את G . لكن אי אפשר פשוט להגיד את G_i לפני θ , שכן הקשר בינהם הוא עקיף ותלוי ב- $f(\theta)$. כדאי להתמודד עם בעיה זו באמצעות log-derivative trick-*trick* (לעתים התהילה זהה נקרא Reinforce). נגיד:

$$g(\theta) = \mathbb{E}_{\pi_\theta}[f(x)]$$

ואז הנגזרת הינה:

$$\nabla_\theta g = \mathbb{E}_{\pi_\theta}[f(x) \nabla \log(\pi_\theta(x))]$$

הוכחה:

$$\nabla_\theta g = \nabla_\theta \mathbb{E}_{\pi_\theta}[f(x)] = \nabla_\theta \int \pi_\theta(x) f(x) dx$$

נחלף את הסדר של הגרדיינט והאנטגרל:

$$= \int \nabla_\theta \pi_\theta(x) f(x) dx$$

כעת נשמש בקשר הבא (המתקיים עבור כל פונקציה $h(\theta)$):

$$= \int \nabla_\theta \log(\pi_\theta(x)) \pi_\theta(x) f(x) dx$$

זהו בדיק שווה לתוחלת הבאה:

$$= \mathbb{E}_{\pi_\theta}[f(x) \nabla \log(\pi_\theta(x))]$$

כעת, במקום לחשב את הגרדיינט של G_i , נוכל לחשב את התוצאה $\mathbb{E}_{\pi_\theta}[f(x) \nabla \log(\pi_\theta(x))]$ על ידי דוגמאות ושערון מונטה קרלו, רק כעת הנגזרת אנו צריכים לחשב **אינה תליה** בפרמטר θ ולכן ניתן לחשב אותה בצורה ישירה. בצורה מפורשת, פונקציית המטרה שלנו הינה:

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[G(\tau)], G(\tau) = \sum_{t=0}^{T(\tau)} \gamma^t R_{t+1}^{(\tau)}$$

שימוש ב-*log-derivative trick* יביא אותנו לביטוי:

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[G(\tau) \nabla \log(\pi_\theta(\tau))]$$

נשים לב שהאסטרטגיה (τ) מורכבת משני אלמנטים – הפעולה שאנו בוחר לבצע, והמצב אליו אני מגע ייחד עם התגמול המתkeletal מצעוד זה. החלק הראשון הוא למרי בשילטה שלו ואנו ידעים אותו, אך החלק השני הוא דינמי ותלוי בסוטופטיות של המערכת, וכן לכארה הוא לא ידוע ולא אפשר לבצע את הגזרה. אך אם נפתח את הביטוי שקיבלו נראה שכל החלקים התלויים בדינמיות של המערכת אינם תלויים ב- θ ויתאפשרו בזירה. ניתן לרשום את האסטרטגיה כמכפלת ההסתברויות של כל הצעדים באופן הבא:

$$\pi_\theta(\tau_i) = P(S_0^i) \cdot \pi_\theta(A_0^i | S_0^i) \cdot P(S_1^i, R_1^i | S_0^i, A_0^i) \cdots \pi_\theta(A_{T^i-1}^i | S_{T^i-1}^i) \cdot P(S_{T^i}^i, R_{T^i}^i | S_{T^i-1}^i, A_{T^i-1}^i)$$

כעת אם נוציא לוג, המכפלה תהופיע לסכום:

$$\log(\pi_\theta(\tau_i)) = \log(P(S_0^i)) + \sum_{t=0}^{T^{i-1}} \log(\pi_\theta(A_t^i | S_t^i)) + \sum_{t=0}^{T^{i-1}} \log(P(R_{t+1}^i, S_{t+1}^i | S_t^i, A_t^i))$$

הביטוי האמצעי מבטא את בחירות הצעדים של המודל, והוא **היחיד** שתלי בפרמטר θ . שני הביטויים הנוספים מבטאים את הדינמיות של המודל שאין לנו מידע כלפיהם, והם אינם תלויים ב- θ , אך בנגזרת לפ' θ הם מתאפסים. עובדה זה בעצם מסקנת יתרון נוספת לבריך של הלוג – מלבד האפשרות לחשב את הנגזרת באופן ישיר, הelog גם מפריד בין הצעדים של המודל לבין הדינמיות הלא ידועה של המודל. לכן בסך הכל נקבל:

$$\nabla_\theta \log(\pi_\theta(\tau_i)) = \sum_{t=0}^{T^{i-1}} \nabla_\theta \log(\pi_\theta(A_t^i | S_t^i))$$

כעת נציב את זה בגדיinant של פונקציית המטריה ונשתמש בדגימות מונטה קרלו:

$$\begin{aligned} \nabla J(\theta) &= \mathbb{E}_{\tau \sim \pi_\theta}[G(\tau) \nabla \log(\pi_\theta(\tau))] \approx \frac{1}{N} \sum_i G_i \nabla_\theta \log(\pi_\theta(\tau_i)) \\ &= \frac{1}{N} \sum_i \left(\sum_{t=0}^{T^{i-1}} \gamma^t R_{t+1}^i \right) \left(\sum_{t=0}^{T^{i-1}} \nabla_\theta \log(\pi_\theta(A_t^i | S_t^i)) \right) \end{aligned}$$

חישוב הגדיinant בדומה זו נקרא בשם הפופולרי **Policy Gradient**. כעת מוחשייבנו את הגדיinant, נוכל להשתמש בשיטות אופטימיזציה סטנדרטיות על מנת למצאו את θ האופטימלי. בקצרה נוכל לסכם את כל התהליך **Reinforcement Learning** בשלושה שלבים:

1. Run the policy and sample $\{\tau^i\}$ from $\pi_\theta(A_t | S_t)$.
2. Estimate gradients: $\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_i \left(\sum_{t=0}^{T^{i-1}} \gamma^t R_{t+1}^i \right) \left(\sum_{t=0}^{T^{i-1}} \nabla_\theta \log(\pi_\theta(A_t^i | S_t^i)) \right)$.
3. Improve the policy using gradient descent/ascent: $\theta \leftarrow \theta_\alpha \nabla_\theta J(\theta)$.

נתובן על הביטוי של הנגזרת שקיבילנו וננסה לנתח אותו. נניח יש זומחה שיודע לצייר trajectories סובים (-) – במקורה נרצה��取 את מה שהוא ייצור וගרים לכך שהסטרטגייה שלו תיציר בהסתברות יותר גבואה כמו שקיבילנו זומחה. בימילים אחרים נתונים לנו מסלולים מוצלחים, נרצה לගרים לאסטרטגיה שלנו "להקוט" את אותם מסלולים. הדרך סטנדרטית לבצע זאת מושמה היא בעדרת maximum likelihood – לחקות את המסלולים האלה ולסוכם אותם, ואז לחפש את המקסימום באמצעות גזירה של הביטוי המתkeletal. באופן שקול ניתן גם לחפש מקסימום עבור הסכום של **לוג המסלולים**, ובכך נרצה לחשב את הביטוי:

$$\sum_{t=0}^{T^{i-1}} \nabla_\theta \log(\pi_\theta(\tau_i))$$

זה בדיק הביטוי השני בגדיinant שראינו קודם. בנוסף אליו יש גם את הגורם הראשון שמשקל כל מסלול בהתאם ל-reward, כך שמסלולים בעלי reward גבוה וairoו מסלולים נמוך יקבלו משקל נמוך. יוצא מכך, שהגדיinant מנשה להביא לכך שמסלולים "מושלמים" יופיעו בהסתברות יותר גבוהה.

השעורי של הגדיinant באציגות הביטוי שראינו הוא אמן חסר הטהה, אך יש לו שנות גבואה. בנוסף, אם כל הדגימות הם בעלי reward חיובי (או כלל בעלי reward שלילי), אז יהיה קשה להבחן איזה צעדים יותר טובים ומה לא כדאי לעשות. נראה שתי דרכי לשיפור היציבות של השימוש בגדיinant:

בסיבותיות (casualty) – כיוון שהצעד a_t שהתרחש בצעד t לא יכול להשפיע על כל הצעדים שקרו לפני כן בסיבותיות את הביטוי, ניתן להחליף את הביטוי $\sum_{t=0}^{T^{i-1}} \gamma^t R_{t+1}^i$ בביטוי שמתיחס רק בצעדים שקרו החל מצעד t , כלומר: $\{a_0, a_1, \dots, a_{t-1}\}$. החלפה זו מורידה את השונות כיוון שהיא גורמת לכך שככל

צעד יורה תליי בפחות איברים אקראיים. הביטוי שהתعلמנו ממנו הוא $\sum_{k=0}^{t-1} \gamma^k \mathcal{R}_{t+1}^k$, וניתן לראות שכאן אין לו השפעה על העטי.

Baseline – ניתן להראות שהוסף של קבוע reward לשיפור מושפע על התוצאה של הגרדיאנט. קבוע מסוים יכול לגורם לכך שנרצה לשפר צעדים אחרים, ואילו קבוע אחר יכול לגורם לכך שנרצה לשפר דואוק צעדים אחרים. קבוע זה הוא למעשה פרמטר נוסף שנitin לשימוש בו, ועל ידי בחירה מושכלת שלו ניתן להפחית את השונות. באופן פורמלי, את הביטוי $\sum_{t=0}^{T-1} \gamma^t \mathcal{R}_t^i$ נחלף בביטוי זהה עם נוספת קבוע:

$$\sum_{t=0}^{T-1} \gamma^t \mathcal{R}_{t+1}^i - b$$

השאלת היא כיצד לבחור את הקבוע b בצויר b ? באופן פשוט ניתן לבחור את הקבוע בתור המוצע של b -rewards. בחירה כזו תביא לכך שהיא שמתוחת למוצע נרצה להוריד את ההסתברות של i ומזה שמלען המוצע נרצה לחזק אותו. בחירה כזו akan מזריד את השונות, אך היא אינה אופטימלית. ניתן לחשב את הקבוע האופטימי בצויר מדעית. הנגזרת של פונקציית המטריה יחד עם האיבר הנוסף הינה:

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [(G(\tau) - b) \nabla \log(\pi_\theta(\tau))]$$

המוצע אינו תלוי ב- b , אך מוביל לרשום את השונות כך:

$$\begin{aligned} Var &= \mathbb{E}_{\tau \sim \pi_\theta} [(G(\tau) - b)^2 \nabla \log(\pi_\theta(\tau))^2] + C \\ &= \mathbb{E}_{\tau \sim \pi_\theta} [G(\tau)^2 \nabla \log(\pi_\theta(\tau))^2] + \mathbb{E}_{\tau \sim \pi_\theta} [b^2 \nabla \log(\pi_\theta(\tau))^2] - 2 \mathbb{E}_{\tau \sim \pi_\theta} [G(\tau) \cdot b \nabla \log(\pi_\theta(\tau))^2] + C \end{aligned}$$

כעת נגזר את השונות לפי b :

$$\begin{aligned} \frac{dVar}{db} &= 2 \mathbb{E}_{\tau \sim \pi_\theta} [b \nabla \log(\pi_\theta(\tau))^2] - 2 \mathbb{E}_{\tau \sim \pi_\theta} [G(\tau) \cdot b \nabla \log(\pi_\theta(\tau))^2] = 0 \\ &\rightarrow b_{opt} = \frac{\mathbb{E}_{\tau \sim \pi_\theta} [G(\tau) \nabla \log(\pi_\theta(\tau))^2]}{\mathbb{E}_{\tau \sim \pi_\theta} [b \nabla \log(\pi_\theta(\tau))^2]} \end{aligned}$$

הביטוי המתקבל הוא אכן המוצע של b -reward, אך מוכפל במסקל התליי בנגזרת של לוג האסטרטגיה. השערון של הביטוי המדוקיק יכול להיות מסובך ורועש, אך לרוב כן משתמשים בשוטוט ב- b -reward המוצע.

אלגוריתם policy gradient policy הינו, כאמור האסטרטגיה שבעזרתה אנו דוגמים מסלולים הינה אותה אסטרטגיה שאנו מוכנים ללביא לאופטימום. כאמור לעיל, אלגוריתמים政策-policy-on, או, א' אפשר "מחזר" דגימות, כלומר בכל נקודת זמן ניתן להשתמש בדגםות שנלקחו מהאסטרטגיה הקיימת באותה נקודת. לעומת זאת האלגוריתם b -off-off-polcy יקירות, נרצה להשתמש באוותן דגימות עברו נקודות זמן שונות, כלומר להפוך את האלגוריתם לחיות-policy. בכך יעשה זאת שلتיקן את השגיאה שנצורת מכך שבקבוקות זמן מסוימת אנו משתמשים בדגםות שנלקחו מאסטרטגיה שכבר השתנה ונעת ההתפלגות של האסטרטגיה שונה. במקרים אחרים – יש לנו דגימות שנלקחו מהתולות מסוימת, ובערותן אנו רצאים לשער תחולת אחרת. זו בעיה מתוחם הסטטיסטייה, והפתרון שלה נקרא

Importance sampling

נניח ואנו רצאים לשערך את $\mathbb{E}_p[f(x)]$, אך אנו לא יכולים לדוגם מהתפלגות q אלא רק מהתפלגות p , כאשר ההנחה היחידה שלנו היא שאם $0 < p < q$ אז גם $0 < q < p$. באמצעות מעבר מתמטי ד' i פשוט ניתן ליצג את התוחלת של p באמצעות התוחלת של q :

$$\mathbb{E}_p[f(x)] = \int f(x)p(x)dx = \int f(x) \frac{p(x)}{q(x)} q(x)dx = \mathbb{E}_q \left[f(x) \frac{p(x)}{q(x)} \right]$$

כעת נדוגם q נושא את התוחלת באמצעות מונטה קרלו:

$$\mathbb{E}_p[f(x)] = \mathbb{E}_q \left[f(x) \frac{p(x)}{q(x)} \right] \approx \frac{1}{N} \sum_i f(x_i) \frac{p(x_i)}{q(x_i)}$$

אם ההתפלגיות (x, q) הן יחסית דומות, אז התוצאה המתקבלת משערכת בצורה יחסית טובת את התוחלת הרציפה. אם זה לא המצב, השונות תהיה גדולה והטואה תהייה לא יציבה. מכל מקום, נוכל להשתמש בرعון זה עברו שערוך הגדר'אנטו:

- For trajectory probability the dynamics does not depend on θ so

$$\frac{\pi_{\theta'}}{\pi_{\theta}} = \frac{\prod \pi_{\theta}(a_t | s_t)}{\prod \pi_{\theta'}(a_t | s_t)}$$

- Can now Compute $\nabla_{\theta'} J(\theta')$ with samples from π_{θ}

- $\sum_{i=1}^N \left(\sum_{t=0}^{T^{(i)}-1} \gamma^t r_{t+1}^{(i)} \right) \left(\sum_{t=0}^{T^{(i)}-1} \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \right) \frac{\prod \pi_{\theta}(a_t | s_t)}{\prod \pi_{\theta'}(a_t | s_t)}$

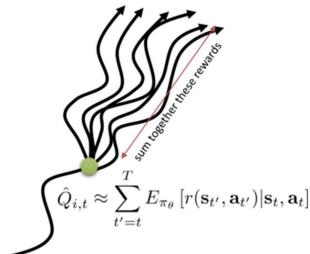
- Can improve with causality

- $\sum_{i=1}^N \sum_{t=0}^{T^{(i)}-1} \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \frac{\prod_{k=0}^t \pi_{\theta'}(a_k | s_k)}{\prod_{k=0}^t \pi_{\theta}(a_k | s_k)} \left(\sum_{k=t}^{T^{(i)}-1} \gamma^k r_{k+1}^{(i)} \right)$

כאמור, שימוש ברעון זה יכול להיות טוב אם ההתפלגיות θ' , θ יחסית דומות. אם הן רחוקות, התוצאה המתקבלת לא בהכרח מושה שערוך טוב, וلين הלמידה לא תהיה מספק טובה. לסייע – אפשר להפוך את האלגוריתם של באופן ישיר (Policy gradient). בפרק זה נציג אלגוריתם המנסה לשלב את שתי הגישות יחד. נתבונן שוב בביטויי של הגדר'אנטו, כאשר בהתייחסות ל-rewards נתיחס רק ל-go-to reward, כפי שהוסבר לעיל:

$$\frac{1}{N} \sum_i \left(\sum_{k=t}^{T^{(i)}-1} \gamma^k \mathcal{R}_{k+1}^i \right) \left(\sum_{t=0}^{T^{(i)}-1} \nabla_{\theta} \log (\pi_{\theta}(A_t^i | S_t^i)) \right)$$

ניתן להבחין כי הביטוי $\sum_{k=t}^{T^{(i)}-1} \gamma^k \mathcal{R}_{k+1}^i$ הוא למעשה דוגמה לש' $\mathcal{Q}^{\pi_0}(s_t, a_t)$, כלומר עבור קווות זמן, ניתן לדגם מסלול באמצעות Action-Value function-הו. ולקבל סכום של rewards. החיבור בדימה זו ונעץ בCKER שיש לה שנות גבואה והיא יכולה להיות מאוד רועשת. כפי שניתן לראות באIOR, תכנו הרבה מסלולים שונים הייצאים מאותה נקודה, וכן הביטוי של ה- $\mathcal{Q}^{\pi_0}(s_t, a_t)$ reward to go.



איור 11.9 מסלולים שונים אפשריים היוצאים מאותה נקודה. המיצוע/התוחלת של הביטוי אמנים מוריד את השונות ל-0, אך כל דוגמה בפניהם היא בעלי שנות גובהה.

הבחנה זו של הקשר בין $\text{go to } b$ ו- $\hat{Q}^{\pi_0}(s_t, a_t)$ מעלה את השאלה האם אפשר להחליפ את בין הביטוי הרוועש $\sum_{k=t}^{T-1} \gamma^k \mathcal{R}_{k+1}^i$ לבין הפונקציה עצמה $\hat{Q}^{\pi_0}(s_t, a_t)$. בשביל לביצוע החלטה זו, יש להוכיח שהיא אכן חוקית ושומרת על תוצאה נכונה, וכי לעתות זאת פונקציית המטרה כרך שתווסף בצורה יותר נוחה. כאמור, פוקנציית המטרה אותה אמן מעוניינים להביא לאופטימום הינה התוחלת של rewards - discount (מכפלים ב-) (factor):

$$\mathcal{J}(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}_{t+1} \right]$$

המשמעות של התוחלת היא לבצע אינטגרל על כל צמד של (s, a) – action-state. כיוון שצמד יכול לחזור על עצמו בזמנים שונים (כלומר בשתי מוקודות זמן t_1, t_2 , $t_1 < t_2$) ניתן ליחסו באותו מצב ולבצע את אותו פעולה), אך למשעה באינטגרל ישם ביטויים שונים שחוורם על עצם. בכך ניתן ליחסו כל ביטוי באינטגרל פעם אחת, ולהכפיל אותו במסקל המתאים למספר הביקורים באותו צמד (s, a) . עם זאת, עדין צריך לזכור שהMOVIES של אותו צמד נבדלים זה מזה – discount factor , כיוון שהמשקל של reward בזמן t_1 שונה מאשר המשקל שלו בזמן t_2 , ויש לקח את ההבדל הזה בחשבון.icut נתבונן על הביטוי המפורש של האינטגרל המתkeletal, כאשר ראשית נחליף את סדר התוחלת והסכימה:

$$\mathcal{J}(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}_{t+1} \right] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\pi_\theta} [\mathcal{R}_{t+1}]$$

התוחלת של כל rewards - discount מרכיב שלם על פה: 1) הסתברות להתחיל במצב s_0 כפוף 2) הסתברות להגיע במצב s בזמן t בהינתן שהתחלנו מ- s_0 כפוף 3) הסיכוי לביצוע פעולה a במצב s שהתקבל בזמן t . באופן פורמלי, הביטוי המתkeletal הינו

$$\begin{aligned} &= \sum_{t=0}^{\infty} \gamma^t \int_S \int_S \int_A p(s_0) p_t^\pi(s|s_0) \pi(a|s) \mathcal{R}(s, a) ds_0 ds da \\ &= \int_S \left(\sum_{t=0}^{\infty} \gamma^t \int_S p(s_0) p_t^\pi(s|s_0) ds_0 \right) \int_A \pi(a|s) \mathcal{R}(s, a) ds da \end{aligned}$$

נסמן את הביטוי שבסוגרים כ- $(s)^\pi$, והמשמעות של $(s)^\pi$ להציג ל gearing s כ- π , כאשר עבור כל t יש להכפיל ב- discount factor המתאים. ביטוי זה נקרא Discount state visitation measure. ולמעשה הוא כולל בתוכו את כל המופיעים של צמד (s, a) , ובכך במקומם לעשות אינטגרל על אותו צמד (s, a) . מספר פעמים, עושים זאת פעמיון נקודות הזמן השונות וככפלים ב- discount factor - discount factor . באופן מפורש:

$$= \int_S \rho^\pi(s) \int_A \pi(a|s) \mathcal{R}(s, a) ds da,$$

$$\rho^\pi(s) = \sum_{t=0}^{\infty} \gamma^t \int_s p(s_0)p_t^\pi(s|s_0)ds_0$$

יש לשים לב שהביטוי הכלול הוא לא בצורה הפוכה של תוחלת – אינטגרל של הסתברות כפול פונקציית צפיפות, אך הוא עדין משקרף תוחלת וכאן ניתן לשער אוות באמצעות מונטה-קרלו. עצה נתבונן על הגדריאנט של פונקציית המטריה. אנחנו רוצים לגזר את הביטוי הבא לפ' θ :

$$\mathcal{J}(\theta) = \int_S \rho^\pi(s) \int_A \pi(a|s) \mathcal{R}(s, a) ds da$$

כל האיברים תלויים ב- θ אך ניתן להראות (שколоויות 10-11 במצגת 7) שהנגזרת תלויה רק ב- $(\pi(a|s), \text{כלומר}:$

$$\nabla_\theta \mathcal{J} = \int_S \rho^\pi(s) \int_A \nabla_\theta(\pi(a|s)) \mathcal{Q}^{\pi_\theta}(s, a) ds da$$

עצה ניתן להיעזר ב-log-derivative trick וללקבל:

$$\nabla_\theta \mathcal{J} = \int_S \rho^\pi(s) \int_A \pi(a|s) \nabla_\pi \log(\pi(a|s)) \mathcal{Q}^{\pi_\theta}(s, a) ds da$$

ואז לשערך את הביטוי באמצעות דגימות מונטה-קרלו:

$$\nabla_\theta \mathcal{J} \approx \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^{T^{(i)}-1} \nabla_\theta \log \pi_\theta \left(a_t^{(i)} | s_t^{(i)} \right) \mathcal{Q}^{\pi_\theta} \left(s_t^{(i)}, a_t^{(i)} \right)$$

הביטוי המתקבל טוב מאוד מביחינת זה שהוא מצליח להפחית את השונות, אך הבעיה בו נעוצה בכך שהאיבר $\mathcal{Q}^{\pi_\theta}(s_t^{(i)}, a_t^{(i)})$ אינו ידוע! אלגוריתם **Actor Critic** בא להתמודד עם בעיה זו, והרעיון הוא לנסוט לשערך את \mathcal{Q} באמצעות \mathcal{Q} במקביל לשערך את הגדריאנט. ככלומר, בתהליך הלמידה אנו מנסים לשערך שני דברים במקביל:

π_θ (Actor) – הගורם המחליט איזה צעד לנ��וט בכל מצב.

\mathcal{Q} (Critic) – הගורם המבצע אבלוציה של \mathcal{Q} על מנת לשפר את הבחירה של ה-Actor.

הurette אגב: לעיל ראיינו שהשימוש ב- Learning-by-Utility – \mathcal{Q} יכול להיות בעייתי בעקבות רציפות בغالל שמרחיב המצביעים הוא איסוטופי ומואוד קשה לפתור את בעיית האופטימיזציה של בחירת action מושלמת בעזרת הביטוי $\arg \max_A \mathcal{Q}_\theta(S, a)$. כאן בעיה זאת אינהנו, כיון שאנו חוננו לא נדרשים לבחור את ה-action \mathcal{Q} באמצעות \mathcal{Q} אלא רק לשפר באמצעותו את האסטרטגיה (או במילים אחרות – אנחנו לא צריכים לפתרו בעיית אופטימיזציה על \mathcal{Q} עבור מרחב מצבים ורים).

האלגוריתם פועל בהתאם לשלבים הבאים. ראשית נאותל את w, θ . עצה נבעץ T איטרציות באופן הבא:

- דוגום מצב התחלתי s_0 מຕוך המצביעים \mathcal{G} ופעולה מຕוך האסטרטגיה $(\pi_\theta(s_0|s_0, \cdot))$.
- עצה כל עוד לא הגיעו למצב הסופי:

- נחשב את המצב החדש s *new* ואת התגמול המתקבל r .

- בסיום, נדגום פעולה חדשה $(\pi_\theta(s|new, \cdot))$.

- נחשב את TD error:

$$\delta = r + \gamma \mathcal{Q}^w(new s, new a) - \mathcal{Q}^w(s, a)$$

- נעדכן את הפרמטרים:

$$\begin{aligned} \theta &\leftarrow \theta + \alpha \mathcal{Q}^w(s, a) \nabla_\theta \log \pi_\theta(a|s) \\ w &\leftarrow w + \beta \delta \nabla_w \mathcal{Q}^w(s, a) \end{aligned}$$

- נעדכן את המצב והפעולה:

$$a = new a, s = new s$$

סואו קוד עבור האלגוריתם:

```

Initialize parameters  $s, \theta, w$  and learning rates  $\alpha_\theta, \alpha_w$ ; sample  $a \sim \pi_\theta(a|s)$ .
for  $t = 1 \dots T$ : do
    Sample reward  $r_t \sim R(s, a)$  and next state  $s' \sim P(s'|s, a)$ 
    Then sample the next action  $a' \sim \pi_\theta(a'|s')$ 
    Update the policy parameters:  $\theta \leftarrow \theta + \alpha_\theta Q_w(s, a) \nabla_\theta \log \pi_\theta(a|s)$ ; Compute
    the correction (TD error) for action-value at time t:
         $\delta_t = r_t + \gamma Q_w(s', a') - Q_w(s, a)$ 
    and use it to update the parameters of Q function:
         $w \leftarrow w + \alpha_w \delta_t \nabla_w Q_w(s, a)$ 
    Move to  $a \leftarrow a'$  and  $s \leftarrow s'$ 
end for

```

11.10 אלגוריתם Actor-Critic. שערור הגדיאנט באמצעות \hat{Q} במקום הביטוי של ה- h . reward-to-go Policy gradient-algorior.

חשיבותו של הדרישת מטרת המבצע בדומה ל-SARSA – \mathcal{Q} – Learning. בינהם הוא אופי האלגוריתם – SARSA הוא אלגוריתם policy-on-policy, כלומר הוא מנסה ללמד את האסטרטגיה האופטימלית בעלי תильות בדוגמאות הוא דוגם את המסלולים, בעוד ש- \mathcal{Q} – Learning מנסה ללמד את האסטרטגיה האופטימלית בעלי תילות בדוגמאות אותן הוא רואה. אלגוריתם Actor Critic רצה לשפר את \mathcal{Q} , כלומר המטרה היא לשפר את האסטרטגייה הנוכחית, וכך ישבץ למידה שהיא שווה עליל, הגדיל את ההבדל.

בדומה ל-*Actor Critic*, גם *Policy gradient* מ intent להויסף על מנת לשפר את תהליכי הלמידה. בינויד לא-*Policy gradient* הוא התוספת היהת של קבוע, אבן גורן להויסף את (s, π^{old}), שהוא תלי' ב- s . אם ה-*baseline* תלי' רק במצב s , אז זה בסדר כי שוכחים מיד (16), אבל אם הוא תלי' גם ב- a , אז זה מוסיף וולמידה *baseline*- c (המשמעותי: $\pi^{\text{old}}(s) = \pi^{\text{new}}(s, a) - V^{\text{old}}(s) + V^{\text{new}}(s, a)$). נקרא $A^{\pi}(s, a)$, *advantage function*- a , ומשמעותה לבחירה של a בתפעע. היא פשוטה – אם הערך של $A^{\pi}(s, a)$ גבוה, אז זה סימן לכך שהבחירה בעצעד a טוביה, ולכן נרצה לחזק אותה. אם לעומת זאת הערך של $A^{\pi}(s, a)$ נמוך, זה אומר שהבחירה לא טוביה ומילא ונרצה להימנע منها. במילים אחרות: (s, π^{old}) הינו הערך הממוצע של המצב s , וביתוי ($s, \pi^{\text{old}}(s)$) הינו הערך של פעולה ספציפית עבור מצב s , ולכן $A^{\pi}(s, a)$ מציין את גורעה ביחס לממוצע של המצב בו אנו נמצאים, ומילא תוכל לדעת האם רצוח אמר לנו עד כמה הפעולה a טוביה או לא.

בשערו (s) Q^{π} ולחשב את (s) π^* ואז להשתמש בו כ-Baseline. על כל הפעולות: (s) $Q^{\pi_{\theta}}$ [$\mathbb{E}_{\pi_{\theta}}[r_t | s_t = s]$] על (s) Q^{π} מתקיים: $a \cdot \phi_s^T(s) = \mathbb{E}_a[\pi(a|s) \cdot Q^{\pi}(s, a)]$, והאסטרטגיה היא אסוציאטיבית: $(\pi(s), \sigma^2, \mu, \rho, \alpha)$ מגדירים $Q^{\pi}(s) = \mathbb{E}_{\pi}[r_t + \gamma Q^{\pi}(s') | s_t = s]$. בחרת הנוסחה הסוגרת זו ניתן להשתמש ב-Baseline (s) Q^{π} ולהשתמש ב- π^* .

מכותם בקשר לערך $r_{t+1} = \gamma V(s_{t+1}, a_{t+1})$, ולפי זה-ה-הנחה: advantage function $V(s_t, a_t)$:

$$A(s_t, a_t) = r_{t+1} + \gamma \mathcal{V}(s_{t+1}) - \mathcal{V}(s_t)$$

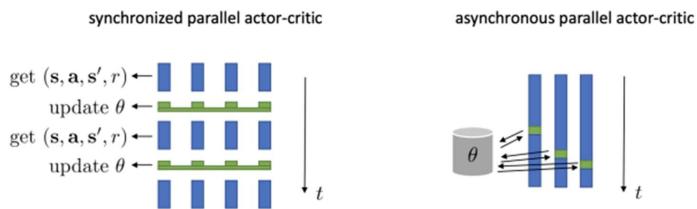
את הביטוי זהה נקבע לגדיאנט ובכך מושך את האסטרטגיה. נשים לב שבביטוי זה תלי' במצב בלבד ולא בפעולה, מה שמקל על הלמידה, כיון שייתור קל ללמידה את (s) π המשיר למצבים בגודל M מאשר את (a), (s) π המשיר למצבים בגודל A . המשמעות הפרקטי של הביטוי היא שימושתיכים על שני מצבים, בווחונם את הפרש הערכים שלהם וועוד the reward . אם הערך חיובי – אז נרצה להזק את המעבר מהמצב הראשון למצב השני, ואם הערך שלילי, אז נרצה למנוע מזה מעבר.

חסר – שקיות 19.

אחת הביעות שיש גם ב- Actor Critic – Learning – Q ו- π גם ב- Learning – Actor Critic יש בין המצביעים השונים, ולכן השונות עדין גבוהה. לעיל ראיינו שבאלגוריתם – Q ניתן להתגבר על

כך בודעת שמירה של N מצבים ועדיין לפ' k מצבים אקראיים מתוך אלה נשמרו (למשל: $N = 1000, k = 10$).
כאן לא ניתן להשתמש בפתרון זה, שכן הلمידה היא policy-טס או אף להשתמש במצבים קודמים,
כיוון שהן דגמו אסטרטגיה שרגעה הא לא רלוונטי. ניתן בכל אופן להשתמש במצבים שדגמו מהה-epochs –
האחרונים, תחת הנחה שהאסטרטגיה לא השתנה הרבה (ובנוסף ניתן לתקן באמצעות IMPORTANCE sampling).

פתרון יותר מוצלח הוא לאמן מספר סוכנים במקביל ולהשתמש במידע שמנוע מכלום, כאשר היתרונו הדגול של שיטה זו הוא שהדgelיות של הסוכנים השוניים הן חסרות קורלציה. שיטה זו תוצאות מועלות ובזמן הרובה יותר מהיר מהשיטות הקודמות שראינו. יש שתי אפשרויות להריץ במקביל – ריצה סינכרונית (הסוכנים רצים במקביל, ולאחר מכן מס'ם epoch מחשבים את הגדריאנט וمعدכנים אסינכרונית) וריצה אסינכרונית (כל פעם שסוכן מס'ם יש לחכות של כל הסוכנים ייימנו. מצד שני הריצה האסינכרונית יכולה להיות פחות מديدة כיון שהיא יכולה לעמוד שולносוכנים שונים תחת אסטרטגייה מעטה שונה (אם הם עדכנו את האסטרטגיה שלהם בזמנים שונים ובאיזהו ערך נקבע בפועל). עם זאת ניתן להניח שהבדלים יחסית לא גדולים, כיון שהסוכנים מעדכנים את האסטרטגיה שלהם בערך באותו זמן).



איור 11.11 מספר סוכנים במקביל בשיטת Actor-Critic. ניתן לבצע את העדכון בזירה סינכרונית – כאשר כל הסוכנים מס'ם epoch מחשבים את הגדריאנט וمعدכנים את האסטרטגיה, או בזירה אסינכרונית – כל סוכן מס'ם epoch מעדכן את האסטרטגיה בהתאם.

לסיום, שיטת Actor-Critic מנsea לשלב עקרונות מתוך Q – Q -Learning reward-to-go Policy gradient – ייחד עם העיבוד הגרפי של הבטי של Q (policy gradient reward-to-go). מהו הרובה פחות רושע. את Q ניתן לשערר במקביל לגדריאנט עצמו, אך לשימוש זאת נדרש ארכיטקטורה מיוחדת אשר אדריך ייש הטיה, וכיון שיש צורך לשערר גם את Q . בדומה ל-Baseline, גם כאן ניתן להוסיף גיבובים (baseline), וריאנו שמקובל להוסיף את Q .

11.4 Model Based Control

בפרק הקודם דיברנו על אלגוריתמים שלא מתעסקים במודל של הבעיה, אלא מנאים ללמידה מודול הניסיון את הערך של כל מצב/מצב-פעולה $(s, a) \rightarrow Q(s, a)$ או את האסטרטגיה האופטימלית עבור הסוכן. בפרק זה מדובר על אלגוריתמים שימושיים ללמידה את האסטרטגיה מודול או הדינמיקה של הבעיה (בין אם ידועים ובין אם הם נלמדת).

יש מספר יתרונות למידה מודול ולא מודול ניסיון בלבד. ראשית, הרובה פעים הלמידה של מודול יכול להיות משגשגת יותר מאשר למידה של $(s, a) \rightarrow Q(s, a)$, מה שהופך את הלמידה לרובה יותר חסכונית. ייקח לדוגמה מבודק פשוט – הבנה של הדינמיקה של הרים הינה יותר פשוטה מאשר למידה של הרים כל משכנתה. בנוסף, למידה של המודול אפשרית exploration – נניח ואנו רצים לאמן סוכן שמבצע ניגוד ניגודית, אם הוא לא ידע את הדינמיקה של הסביבה אלא ינסה ללמוד רק על בסיס הניסיון, הוא יהיה חייב לבצע פעולות שליליות (למשל לדודס אנשים ולעשות תאונות) על מנת לקבל משוב חיובי, וכך ללמידה שיש להימנע מפעולות אלה. אם לעומת זאת אנו ידועים את המודול והסבירה, ניתן ליצור סימולציה ובה לבחון את המצבים השונים ורק לאמן את הסוכן.

עם זאת, ישנו גם חסרונות בלמידה של המודול. ראשית, הרובה פעים יש פערים בין המודול לבין העולם האמיתי. נניח ואימנו סוכן של רכב אוטומטי בסביבה ורלוונטי, המ עבר לעולם האמיתי אנו טרוייאלי כי הסימולציה פשוט לא מספיק דומה לעולם האמיתי (למשל – רגולאיות תנועות הקולט של הסוכן בסימולציה הרבה יותר גבוהה מאשר מצלמה אמיתית המותקנת על רכב). בנוסף, למידה של מודול מושיפה מוקור נסוף של שגיאה – נניח ומנסים לשערר מודול ועל בסיס המודול רצים ללמידה אסטרטגיה, אך יש פה שני שלבים של למידה וכל אחד מהם יכול לגרום לשגיאה.

11.4.1 Known Model – Dyna algorithm

נתייחס למקרה היותר פשוט, בו המודל ידוע לנו מנסים באמצעותו ללמידה אסטרטגיה אופטימלית. בפרק המבוא ראינו כיצד ניתן להשתמש ב-policy iteration Tabular MDP עבור מרחיב מצבים לא גדול על מנת ללמידה את האסטרטגיה האופטימלית, ובפרק זה נרצה להתמקד במרקם בהם פתרון זה לא מספק טוב.

הרעישן הפשט בוחר בו ניתן להשתמש להשתמש הוא ליצר סימולציה, כולם לחת את המודל הידוע, ולאחר מכן סוכן אליו שהוא נמצא בעולם אמיתי. למידה של אסטרטגיה או של $(s, a) \rightarrow Q(s, a)$ באופן זהה לרוב תהיה צולга שימושית מאשר למידה שלהם ללא עזרת המודל, כיוון שנוצר הרבה חיות איטרציות והשימוש במודל הידוע מסיע להוכיח את הלימידה. עם זאת, הבעיה היא שיש פער בין הסימולציה לבין העולם האמיתי, מה שגורם אתגר כפול. א. סימולציה לא מגיעה לרמת דוק מספק טובה בתיאור העולם האמיתי, אך המודלה לא מספק איפותית. ב. נניח ויש לנו אפשרות ללמוד על העולם האמיתי, אך גם אם יש שגיאה – היא לרוב תהיה קטנה מאוד. בסימולציה לעומת זאת, שגיאה כזו יכולה להצטבר ולגרום לסוכן לשגיאות גדולות.

בכדי להתגבר עלאתגרים אלו ניתן **לשלב** בין Model free ו-Model Based, וcutout נציג אלגוריתם בשם **Dyna** שעושה בדיק את השילוב הזה. אלגוריתם זה משלב ידי' משני מקורות –>Dataת אמיתית ואתה המגעים מיסימולציה, כאשר הוא פועל באופן איטרטיבי לפי השלבים הבאים:

- ראשית מקבלים>Dataת אמיתית, ומפרים באמצעותו שני דברים – מודל שעליו מושחת הסימולציה וסוכן שמואמן בכלים של model free.
- לאחר מכן מייצרים>Dataת מהסימולציה, ומשתמשים בו בשביל לשפר את הסתוכן.

מבצעים את השלבים האלה שוב וככה מփררים במקביל גם את הסימולציה וגם את הסוכן (שלמוד על בסיס>Dataת מהסימולציה). באופן זהה אנחנו מפרים את הסימולציה כל הזמן, ואנו גם יש בעיה בסימולציה והוא לא זזה בדיק למודול של העולם האמיתי, אנחנו לא מתעלמים ממנה אלא מנוטים לתזקן אותה. בנגדוד לSIMULACRA פשוטה שלא משתמשת>Dataת מהעולם האמיתי לא רק מtaboosted על המודל הידוע, כאן אף לא זונח את>Dataת האמיתית אלא עושים לו אוגמנטיות בעזרת הסימולציה (שב עצמה גם הולכת ומשתפרת). נניח מקרה פשוט של מודל דטרמיניסטי ובחירה של אלגוריתם – Q – Learning הסוכן, האלגוריתם יפעל לפי שלדים הבאים (כמובן שניתן להרחב בצהורה טריוויאלית לכל מודל ולכל אלגוריתם (Model-Free):

```

Initialize  $Q(s, a)$  and  $Model(s, a)$  for all  $s \in \mathcal{S}$  and  $a \in \mathcal{A}(s)$ 
Do forever:
    (a)  $S \leftarrow$  current (nonterminal) state
    (b)  $A \leftarrow \epsilon\text{-greedy}(S, Q)$ 
    (c) Execute action  $A$ ; observe resultant reward,  $R$ , and state,  $S'$ 
    (d)  $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$ 
    (e)  $Model(S, A) \leftarrow R, S'$  (assuming deterministic environment)
    (f) Repeat  $n$  times:
         $S \leftarrow$  random previously observed state
         $A \leftarrow$  random action previously taken in  $S$ 
         $R, S' \leftarrow Model(S, A)$ 
         $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$ 

```

איור 11.12 אלגוריתם Dyna. שילוב של למידה מיסימולציה המבוססת על מודל ידוע יחד עם סוכן שלמוד בעזרת אלגוריתם Model-Free.

מלבד העבודה שתור כד' הלימידה אנו מפרים את המודל, גם>Dataת שמייצר הוא יותר לרלונטי ובולע ערך ללמידה – לא סתם מוגלים מבנים אקראיים, אלא המיצבים שהסוכן רואה קשורים לאסטרטגיה שלמודה על בסיס>Dataת המהעולם האמיתי. באופן זהה יש שימוש כפול ו"מצח" יותר של>Dataת מהעולם האמיתי – הוא גם מסיע לבנות את המודל וגם מסיע לסוכן להשתפר.

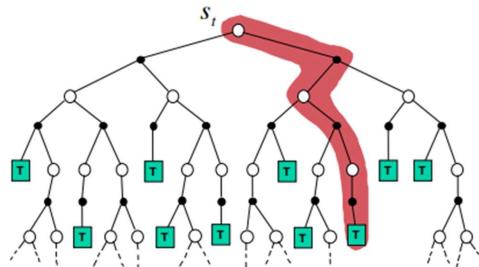
גרף ודוגמה – שקופיות 12, 11. שינוי מודל למודול לא נכון – שקופיות 13 (מודול קבוע לא היה מצליח להשתפר לאחר השינוי).

11.4.2 Known Model – Tree Search

כאמור, אלגוריתם מה-Dyna נועד בידעת המודל על מנת לאמן סוכן בעזרת אלגוריתם של Model-Free. יצא שאנו הסוכן אינו תלי במודול, אך עדין האסטרטגיה אותה אנו מנסים ללמידה היא גלובלית, כלומר מנסה לדעת מה לעשות בכל סיטואציה אפשרית. עבור משימות מאוד מורכבות, כמו למשל משחק שחמט, זה עדין לא מספק, כיוון שאסטרטגיה כללית עבור כל המצבים אינה אפשרית ללמידה עקב ריבוי המצבים. בכדי להתמודד עם בעיות מורכבות

ניתן להפוך את הפתרון להיות **לאקלוי**, כלומר להסתכל על המצב הנוכחי בלבד ולנסות ללמידה באמצעות הסימולציות מה הפעולה שתביא את התוצאה הרצויה.

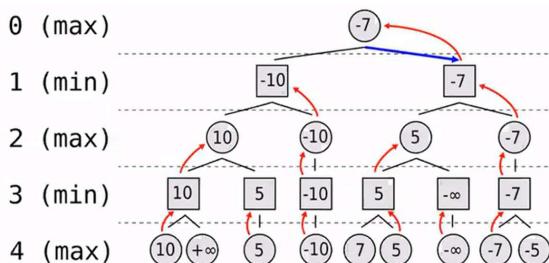
הרעיון של הלמידה הלאקלואלית – חיפוש הפעולה הרצויה עבור מצב נתון – קשור לתהום רחוב שנקרא **Forward Search** (ງ' שבעיה זו קשורה להרבה תחומים ולא רק ל-RL), ופהណון בערך בהמה שקשורה לחיפוש באמצעות אלגוריתם של RL (RL). תחום זה בא לתהמודד עם מצבים בהם הbhaya הכללית היא מאוד מורכבת, אך אנו מנסים לפחות את-ה_bhaya המסתכלת רק על המצב הנוכחי ומה שמסתעף ממנו. ניתן לדמות את הסביבה לעץ בו כל צומת הוא מצב והוא מתפצל למצבים נוספים לפי הפעולות השונות האפשריות באותו מצב. באמצעות הסימולציה נבדוק כל מיי הפעולות שונות מאותו מצב, ונוכל לבחון מי מביניהם הטובה ביותר.



איור 11.13. Forward Search. חיפוש הפעולה הרצויה עבור מצב נתון. כל צומת בעץ מייצג מצב, כאשר מצבים ט�ים (terminal state) מוצגים על ידי T. סימולטור המבוסס על מודל ידוע יכול לעזור לבחון את הפעולות השונות.

כאשר חלק מהbhaya יש גם ייריב שהסטרטגייה שלו אינה ידועה, יש כמה דרכיין איך להתייחס להתנוגות שלו, כאשר הנפוצות שבין הן ליחסו ליריב את אותה אסטרטגייה כמו של הסוקן (Self-play) או ליחסו אליו את האסטרטגייה האופטימלית (Minimax), כאשר יש להגדיר במידק מהו אוטה אסטרטגייה אופטימלית. אם נניח שמדובר במצבים של משחק סכום-אפס (בهم התגמול של שני המשתתפים הוא קבוע), אסטרטגייה אופטימלית מניחה שהיריב יקטם בפעולות שימקסמו את התגמול שלו וימערו עד כמה שנitin את התגמול של הסוקן. במקרה זה, הסוקן ריצה לבורר בפעולת שתמקסם את התגמול שלו worst case, כלומר בחר את הפעולות הטובות ביותר עבורו.

ה_bhaya שעולה מיד היא שבפועל אי אפשר להתחיל ממצב מסוים ולבדוק את כל הפעולות עבור כל עומק אפשרי של העץ. לכן יש להגדיר עומק חיפוש מקסימלי, ואך כל מצב שהגענו אליו שמנצאות אותו עומק יקבל ערך לפי נסחה מסוימת מראש (למשל – בשחמט ניתן לחתם על מצב לפי שווים של כל הסוקנים הנמצאים באותו רגע על הלווח פחוות שווים של כל היריב). נתבונן על דוגמה של עץ maxmin בעומק 4 בעקבותworst case-הו של היריב.



איור 11.14. בעומק 4. העיגולים מסמנים את המצבים בהם תור היריב, והרביעיים מסמנים את המצבים בהם תור הסוקן. החיצים האדומים מסמנים את הפעולות האופטימליות עבור היריב, ואם מינחים שהיריב יבחר בין.

הסוקן מתחילה במצב בשורה 0, ובאזור סימולציה בודק את כל הפעולות עד עומק 4. בשלב ראשון הסוקן משער במערכות הסימולציה רק את הערכים של הצמתים בעומק המקסימלי. אם זה מצב טרמיינלי (כלומר המשחק הסטיים), הערך בעומק זה יהיה הערך שהתקבל בסימולציה, ואם המצב אינו טרמיינלי אז הערך נקבע לפי היריסטיקה קבועה מראש. לאחר מכן הסוקן הולך ומשקל את כל הצמתים בעומק הקודם תחת הנחתה worst case – כלומר מינחים שהיריב יבחר את הפעולות הגדועה עבור הסוקן (מוסמן בחיצים האדומים). ככה למשל ה策מת

השמאלי בעומק 3 קיבל את הערך 10, כי אנו מניחים שבעצם זה הסוקן יבחר לילכת שמאליה. בדומה, ה策מתה הימני בעומק 3 קיבל את הערך 7, כי אנו מניחים שהוא יבחר לילכת שמאלה באותו צמתה. כך נותנים ערכים לכל השורה הזאת, כאשר הכל הוא סלל צמות מקובל את הערך **מג'ימיני**, מבן כל אפשרויות הפעzel שלו.icut ערכים לתת ערכים לשורה בעומק 2, כאשר כאן הכל הוא הפור – כיוון שהטור הוא של הסוקן, נניח שהוא יבחר בפעולתו היכי טוביה, ולכן כל צמות יקבל את הערך המקסימלי מבין הפעולים האפשריים שלו. למשל הצמתה השמאלי בעומק 2 מקבל את הערך 10, כיון שהוא הערך של ההתוצאות היכי טוביה שהסוקן יכול לבחור. באופן זה עולמים למיניהם ומילאים את כל ערכי העץ עד השורש, עד שמקבלים ערך עבור המצב הנוכחי בו אנו נמצאים.

כמה העורות חשובות:

1. יתקן והיריב יעשה טעויות ולא יבחר את הצעדים היכי טובים עבורה, אז בפועל נקבל ערךآخر למצב בו אנו נמצאים, אבל השיטה הזרירה ביצורו היא להניח שהיריב הינה אופטימלי ועל בסיס זה לתת ערכים לצמותים ולמצב הנוכחית. לאחר שבירצינו את הפעולה היכי טוביה בהתאם להערכה הנוכחית, היריב בתוויה בוחר צעד ומתקבל מצב חדש, ואז הסוקן שוב יבצע סימולציות על מנת לחשב את כל ההסתעפויות הקשורות נזודה ולתת ערך למצב החדש שתתקבל.
2. כיון שהעץ גדל אקספוננציאלית, העומקים אותם ניתן לחשב הם לא גדולים, ולכן יש חשיבות להיריסטיקה שקובעת את הערכים עבור הצמותים בעומק המקסימלי שלא קיבלו ערך בסימולציה עצמה (כלומר מצבים שאינם טרמינליים).
3. במשחקים בהם יש גם גורם של מזל (כמו למשל הטלת קוביית), ניתן להוציא שכבת של nodes chance בין כל שתי שורות בעץ, כאשר החישוב יהיה לשכבות החדשות הימין בעדרת התוחלת של הטעאה שלן. עץ זה נקרא **Expectiminimax Tree**.

אלגוריתם חיפוש זה עובד טוב עבור מגון בעיות, אך הוא עדין נתקל בבעיות עבור אטגרים בעלי מרחב מצבים גדול מאוד, כמו למשל המשחק Go. במקרים כאלה מספר הפעולים גדול מידי בשבל שיחיה בר חישוב, וממלא לא ניתן לבחון את כל ההסתעפויות עבור עומק מוגבל. במקרה כזה, ניתן שאל ואפשר לחשב ב佗חה ערכי אוזמומיים. לכן, במקרים לבדיקת את כל ההסתעפויות, ניתן להיעזר בסימולציה ולחזור רק חלק מהן, ועבור כל סימולציה בלבד נלעומך שההתוצאה תהיה אמיתית ולא תלויה בהיריסטיקה. כמובן שבחירה ההסתעפויות לא נעשית באופן אקראי, אלא היא מבוססת על מדיניות למדת של שני היריבים. באופן זה בהתחלה היריב היא קללה כיון שליריב אין אסטרטניה טוביה, אך ככל שאינו משתפר כך גם היריב משתפר ובעיה הולכת ונעשית קשה, ומילא ניתן להשתרע עוד ועוד. אלגוריתם חיפוש זה נקרא **Monte-Carlo Search**, וניתן לנaucו אותו באופן פורמלי כך:

נניח ונתונה אסטרטgia ההתחלתית π שהיא מספיק טוביה. נתן מצב t , ורוצים לדעת מה הערך של כל פעולה a אפשרית. דוגום K מסלולים עברו כל פעולה a אפשרית, נעדיר בסימולציה i לבצע את המסלול עד הסוף, ונחשב את תוחלת הרוח של כל סימולציה. באופן זה, עבור המצב t הערך של פעולה a יהיה:

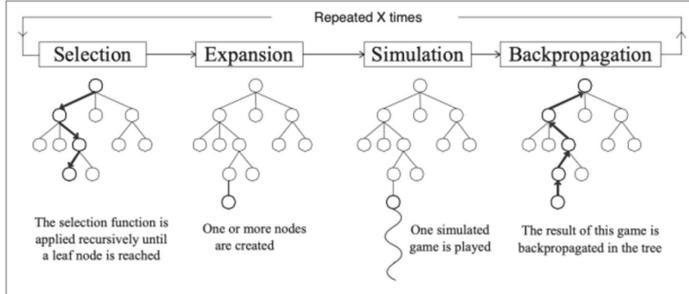
$$Q^\pi(s_t, a) = \frac{1}{K} \sum_k \sum_i \gamma^i R_{t+i+1}^k$$

לאחר שיחסבנו את הערך של כל הפעולות, נבחר את הפעולה בעלת הערך היכי גבוה:

$$\max_a Q^\pi(s_t, a)$$

יש להעיר שכך שאלגוריתם יעבוד, צריך שההסטרטgia ההתחלתית π תהיה ברמה סבירה, כיון שם היא גורעה, השיפור יהיה מאד איטי ולא מספיק.

ניתן לקחת את הרעיון הזה ולהריב אוטו לחיפוש על עץ כדי שיתארו קודם. כאמור לעיל, האטגר המרכז' בחיפוש על עץ נועז בכך שהוא גדל אקספוננציאלית ולא ניתן לחשב באופן יעיל בעיות עם מספר מצבים גדול מאוד. בצד' להתגבר על כך ניתן להשתמש באלגוריתם (MCTS) **Monte-Carlo Tree Search**, שהוכיח את עצמו כמואוד חזק ואמיל לביעיות גדולות מאוד. אלגוריתם זה למעשה משלב בין שני חלקים שככל אחד מהם בפועל עצמו לא מספיק טוב – מצד אחד יש לנו אסטרטגיה תורתית, אך היאBINOMIAL ולא מספיק איקונית, ומצד שני יש לנו סימולציה המבוססת על ידיעת המודל, אך היא אינה יכולה לבדוק את כל האפשרויות. הרעיון של MCTS הוא לקחת אסטרטגיה בינויות ולשפר אותה באמצעות בדיקה מושכלת של העתיד – הרחבה של העץ לא נועשית באמצעות בוחינת כל אפשרויות, אלא באמצעות בחירה של כמה אפשרויות שנראות סבירות ביחס לאסטרטגיה הנוכחית. האלגוריתם ניתן לתיאור באמצעות ארבעה שלבים, שחוורים על עצםם באופן איטרטיבי a פעולות a :



איור 11.14 אלגוריתם Monte-Carlo Tree Search: 1. בחירת הצומת הבא אותו רצים לבחון. 2. הרחבות העץ מאותו צומת. 3. הריצת סימולציה מאותו צומת עד הסוף (=העה למשחק טרמייל). 4. נתינת ערך לכל צומת מהצומת הנבחר ואחוריה עד השועש.

כאמור, האלגוריתם מורכב מארבעה שלבים:

1. בחירה (Selection) – ראיית יש לבחור את המצביע s עבורו אנו רצים לבדוק מה הפעולה הכי טובה במצב זה. לשם כך נבצע סדרה של צעדים עד שנגיע למצביע s , כאשר את הצעדים ינית לבחור על בסיס ה-score שלהם ואפשר גם לשלב רנדומליות עבור exploration כפי שנראה בהמשך.
2. הרחבה (Expansion) – משלגנו לאותו מצב s ונרחיב אותו לצומת נוספת על ידי בחירה של אחת מהאפשרויות האפשריות. את הפעולה ינית לבחור באופן רנדומלי או על בסיס האסטרטגיה הנתונה (שאמור היא ורואה מואופטימלית).
3. הריצת סימולציה (Simulation) – כעת נרים סימולציה מהמצוב החדש עד הסוף וنبדק כיצד הסטיים המשחק. בהתאם לתוצאות המשחק ינית לאותו מצב חדש ערך.
4. נתינת ערכים לצומתים (Backpropagation) – נחלחל אחורה את הערך שהתקבל ונעדכן את כל הצומתים שנבחרו. אם למשל המשחק הסטיים בничחון, אז נעדכן את ערך הצומת החדש (שהרחוב את העץ $-1/1 = 1/1$) (=ניצחון אחד מתוך משחק אחד). בהתאם נעדכן את כל יתר המצביעים שעברנו בהם – אם למשל השורש ממנו ייצאנו היה בעל יחס של 4/6, אז נעדכן אותו ל-3/5.