

המחלקה להנדסת תוכנה

ניתוח תמונות רפואיות

Deep Learning for Medical Images

חיבור זה מהווה חלק מהדרישות לקבלת
תואר ראשון בהנדסה

מאת
גיא ביטון
טל שירן

יוני 2017

כ"ז בסיוון, תשע"ז
המחלקה להנדסת תוכנה
פרויקט גמר – תשע"ז

ניתוח תמונות רפואיות
Deep Learning for

Medical Images

חיבור זה מהווה חלק מהדרישות לקבלת
תואר ראשון בהנדסה

מאת
גיא ביטון
טל שירן

מנחה אקדמי: שפנייר אסף
רכז הפרויקטים: ד"ר ראובן יגל
אישור: _____
תאריך: _____
אישור: _____
תאריך: _____

הצהרה

העבודה נעשתה בהנחיית מר שפנייר אסף,
המכללה האקדמית להנדסה ירושלים - עזריאלי

המחלקה להנדסת תוכנה.

החיבור מציג את עבודתנו האישית
ומהווה חלק מהדרישות לקבלת תואר ראשון
בהנדסה.

תודות

ברצוננו להודות למר אסף שפנייר, על העזרה, על
ההנחיה המקצועית, על האתגר שניתן לנו ועל
הסבלנות הרבה שניתנה במהלך הפרויקט.

תקציר

סיווג איברים אוטומטי היא בעיה חשובה ומאתגרת עבור ניתוח תמונות רפואיות. בשנים האחרונות, פותחו טכניקות לבעיות סיווג (Classification) שונות בתחומים רבים, ובפרט בתחום הרפואה. רוב הטכניקות מבוססות על רשתות נוירונים מלאכותיות ובפרט רשתות נוירונים מלאכותיות בעלות שכבות (ConvNets), אשר טובות יותר בניתוח של תמונות.

בפרויקט זה, אנו נחקור גישה לפתרון עבור סיווג איברים מתוך סריקות CT תוך שימוש ברשתות נוירונים מלאכותיות.

מרכז הפרויקט מתרכז בחקירת בעיית הסיווג על ידי רשתות נוירונים. בהינתן מאגר נתונים של סריקות CT, המטרה היא לבנות מערכת אשר טוענת את הסריקות כך שהמערכת תדע לנתח אותן ולקבל מהן מאפיינים (Features) לשם מטרות הסיווג.

נפרט על שלבי הכנת המידע והשלבים בבניית רשת הנוירונים, נפרט על הרשת שיצרנו, על תהליך אימון הרשת ועל התוצאות שקיבלנו, כמו כן נפרט על הקשיים שעלו בשלבי הפיתוח.

- [תוכן עניינים](#)
- [מילון מונחים, סימונים וקיצורים](#)
- [מבוא](#)
- [מסגרת הפרויקט](#)
- [תיאור הבעיה](#)
 - [דרישות ואפיון הבעיה](#)
 - [הבעיה מבחינת הנדסת תוכנה](#)
 - [תיאור הפתרון](#)
 - [מהי המערכת?](#)
 - [תהליכים ונתוני המערכת](#)
- [תיאור הפתרון המוצע](#)
 - [תיאור הכלים המשמשים לפתרון](#)
 - [הקדמה כללית למימוש הפתרון](#)
 - [תיאור המערכת שמומשה](#)
- [תוכנית בדיקות](#)
- [סקירת עבודות דומות בספרות והשוואה](#)
- [מסקנות מהמימוש ומהפרויקט](#)
- [רשימת ספרות \ ביבליוגרפיה](#)
- [ערכים והגדרות](#)
- [טבלת סיכונים](#)
- [רשימת טבלת דרישות \(User Requirement Document\)](#)

מילון מונחים, סימנים וקיצורים

- CT - שיטת הדמיה (Imaging) רפואית לא-פולשנית המשתמשת בטומוגרפיה הנוצרת על ידי עיבוד של מחשב.
- למידה חישובית (Machine Learning) - תת תחום במדעי המחשב ובבינה מלאכותית העוסק בפיתוח אלגוריתמים המיועדים לאפשר למחשב ללמוד.
- למידה עמוקה (Deep Learning) - תת תחום של למידה חישובית, המבוסס על אלגוריתמים אשר מטרתם ליצור גרף בעל שכבות מרובות. כל קדקוד בגרף מדמה את פעולות הניורון הבודד (Perceptron) ולכן הגרף נקרא רשת ניורונים (Neural Network).
- פרספטרון (Perceptron) - מודל מתמטי המחקק את פעולת הניורון הבודד.
- רשת ניורונים (Neural Network) - רשת ניורונים היא גרף הבנוי מלפחות 3 שכבות (2 עבור שכבת כניסה ויציאה ולפחות אחת של חישובים). השכבות שאינן כניסה או יציאה נקראות Hidden Layers. כל שכבה בנויה ממספר כלשהו של קדקודים המדמים את פעולת הניורון הבודד (פרספטרון).
- רשת ניורונים עם שכבת קונבולוציה (Convolution Neural Network - CNN) - רשת ניורונים המחקק את פעולת קליפה הראייה במוח בו מעובד מידע חזותי במערכת הראייה. הרשת משתמשת בפעולה המתמטית קונבולוציה בין כל שכבה ושכבה, ומכאן שמה. רשת זו טובה בעיקר ללמידה על תמונות.
- שכבות נסתרות (Hidden Layers) - שכבות אשר הפלט שלהם משמש לקלט של שכבה אחרת. השכבות נקראות כך מאחר והן לא הפלט הסופי של רשת ניורונים ולכן הן "נסתרות". ניתן לומר שכל שכבה שאינה כניסה (Input) או יציאה (Output) של רשת ניורונים היא Hidden Layer.
- פונקציית אקטיבציה (Activation Function) - פונקציה המגדירה מהו הפלט של פרספטרון כתלות בקלט.
- פונקציית הפסד (Loss Function) - פונקציה המחשבת את ההפסד (loss) של רשת הניורונים. על סמך פונקציה זו ניתן גם לחשב את הדיוק (Accuracy).
- Overfitting - מצב בו ביצועי הרשת טובים מאוד בשלב האימון כתוצאה מחוסר גיוון במידע. פתרון אפשרי למצב זה הוא להגדיל את כמות המידע.
- פאץ' (Patch) - חתיכה בגודל מוגדר מהתמונה.
- Mini Batches - טכניקה אשר מחלקת את המידע לחלקים ומאפשרת הכנסה של כמות מידע מוגדרת וקטנה יותר לרשת הניורונים.
- AWS - שירותי הענן של Amazon המאפשרים הקמה של מחשב ענן עבור אימון של רשתות ניורונים (Amazon Web Services).

- Segmentation - סימון של איבר על גבי סריקת CT המתבצעת ע"י רדיולוג.
- Segmentation Ratio - אחוז הכיסוי של פיקסלים המסמנים איבר על גבי סריקת CT מתוך פאץ' של סיגמנטציה.
- Standard Deviation - פרמטר הקובע את סטיית התקן של התפלגות נורמלית. בא לידי שימוש באתחול המשקולות. בקיצור נקרא stddev.

מבוא

בעשורים האחרונים חלה התקדמות דרמטית בתחום הדימות (Imaging) בשל ההתקדמות הטכנולוגית המרשימה בתחום המחשוב. קיימים מספר אמצעים המאפשרים לקבל דימות של איברי הגוף כמו למשל, X-RAY, CT, MRI, ULTRA-SOUND ועוד.

מאז שנות ה-70, ישנו שימוש נרחב בתצלומי CT. למכשיר ה-CT הראשון היה צורך בכחצי שעה לקבלת סריקה בודדת, בעוד שכיום, בשניות בודדות, ניתן לקבל סריקה של הגוף כולו - מהראש עד אצבעות הרגליים ובפרוסות של פחות ממחצית המילימטר. בסוף שנות ה-90, המהפכה הגיעה לשיאה, עם כניסתם לשוק של מכשירי ה-CT הרב פרוסתיים (תלת מימדי), מכשירי ה-CT הראשונים הפיקו תמונה אחת תוך 30-40 דק', הרי שהמכשירים החדשים והמתקדמים מצליחים להפיק עשרות תמונות, בתוך פחות משנייה.

תצלום CT מאפשר לקבל תמונה תלת ממדית של פנים הגוף בתהליך לא פולשני. הרדיולוג משתמש במחשב המעבד את הנתונים ויוצר הדמיה תלת ממדית של האזור המצולם על ידי חיבור וניתוח של כל התמונות, בנוסף הרדיולוג מסמן על גבי התמונה אזורים רלוונטים וחשובים לזיהוי בעיות.

מסגרת הפרויקט

פרויקט זה מבוצע תחת הנחייתו של מר אסף שפניר, פרויקט הגמר נעשה במסגרת לימודי תואר ראשון בהנדסת תוכנה ב-"עזריאלי" - המכללה האקדמית להנדסה בירושלים.

הפרויקט מבוסס על פרויקט גמר מהטכניון, אשר מתעסק בניתוח תמונות רפואיות. הפרויקט הנ"ל יכתב בספריה שונה במטרה להקים תשתית סקלבילית לקוד, ולשפר את זמני הריצה והדיוק. כמו כן בפרויקט זה נרחיב את המחקר לסיווג איברים נוספים.

תיאור הבעיה

תהליך פענוח בדיקת דימות הינו מורכב ודורש זמן וניסיון רב. כחלק מתהליך זיהוי בעיה רפואית מתוך סריקת CT, דרוש מומחה שיודע לקרוא את הסריקה ולסווג מתוכה את האיברים. סיווג האיברים היא פעולה רגישה ועשויה לקחת זמן רב גם אם היא מתבצעת על ידי מומחה בעל ניסיון. כמו כן ישנו מחסור תמידי בתחום פענוח הסריקות, ולכן עולה צורך לפיתוח פתרון לסיווג איברים מתוך סריקות CT באופן אוטומטי, מהיר, בטוח ומדויק.

דרישות ואפיון הבעיה

מתיאור הבעיה עולה כי נדרשת מערכת אשר יודעת לסווג איברים מתוך סריקת CT, באופן מדויק ומהיר. הבעיה העיקרית מתחלקת לשני חלקים עיקריים:

1. ייצוג המידע (סריקות CT) בצורה שניתן לנתח ולעבד - כדי שניתן יהיה לחקור את המידע, יש להמיר את הסריקות למטריצות, שעליהן ניתן לבצע חישובים.
2. ניתוח המידע וסיווג איברים מתוכו - יש לבנות מודל חישובי שיהווה את אלגוריתם הלמידה של המערכת. תהליך למידת המידע הוא תהליך ארוך. לאחר שהמודל יגיע לאחוזי דיוק גבוהים (יותר מ 90%), השערתנו היא שהמודל יאפשר לבצע סיווג איברים מתוך סריקות CT בדיוק גבוה.

הדרישה העיקרית היא סיווג איברים בדיוק גבוה, כלומר, המערכת תקבל בכל פעם חלק מהסריקה ותוציא כפלט האם זה חלק מהאיבר הנדרש כלומר סיווג החלק שהרשת קיבלה.

הבעיה מבחינת הנדסת תוכנה

הבעיה מצריכה תכנון מורכב של כל האלמנטים במערכת, הכולל את שלב הבנת המידע, הכנה ונרמול המידע, בניית מודל המבוסס על רשת נוירונים מלאכותית, מציאת הפרמטרים הנכונים של הרשת, ניתוח של הגודל הנכון של הרשת (מספר השכבות הדרושות), שימוש ברשת נוירונים ללמידת המכונה, ניתוח התוצאות והסקת המסקנות. המערכת דורשת תכנון אנליטי ויסודי על מנת שתהיה מודולרית, קלה לתחזוקה וכן סקלבילית כדי שתוכל להתרחב לזיהוי איברים נוספים.

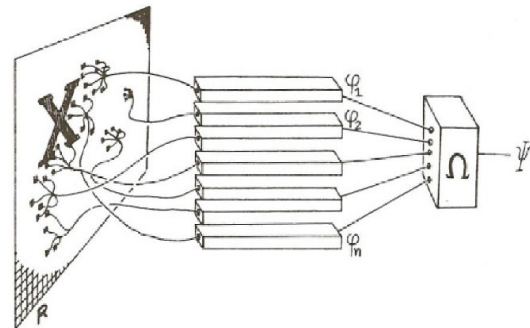
יש לבחור Framework אשר יוכל לענות על דרישות המערכת, וכן מציע אפשרות לבדוק את המערכת. במקרה של רשתות נוירונים הדרך העיקרית לבדוק את המערכת היא באמצעות המדדים Accuracy ו-Loss, שמודדים את ביצועי רשת הנוירונים. כמו כן, יש צורך לבצע ויזואליזציה של תמונות, לצורך ביצוע Debugging על התהליך.

הפרויקט נעשה בשפת Python וכן החלקים השונים של המערכת מחולקים בהתאמה גם בקוד (פונקציות להכנת המידע בקובץ נפרד, רשת הנוירונים בקובץ נפרד וכדומה). בנוסף, גם תכנית הבדיקות בפרויקט זה מסתמכת על מדדים שניתן להוציא על סמך ביצועי רשת הנוירונים, כפי שיפורט בסעיף תכנית בדיקות.

תיאור הפתרון

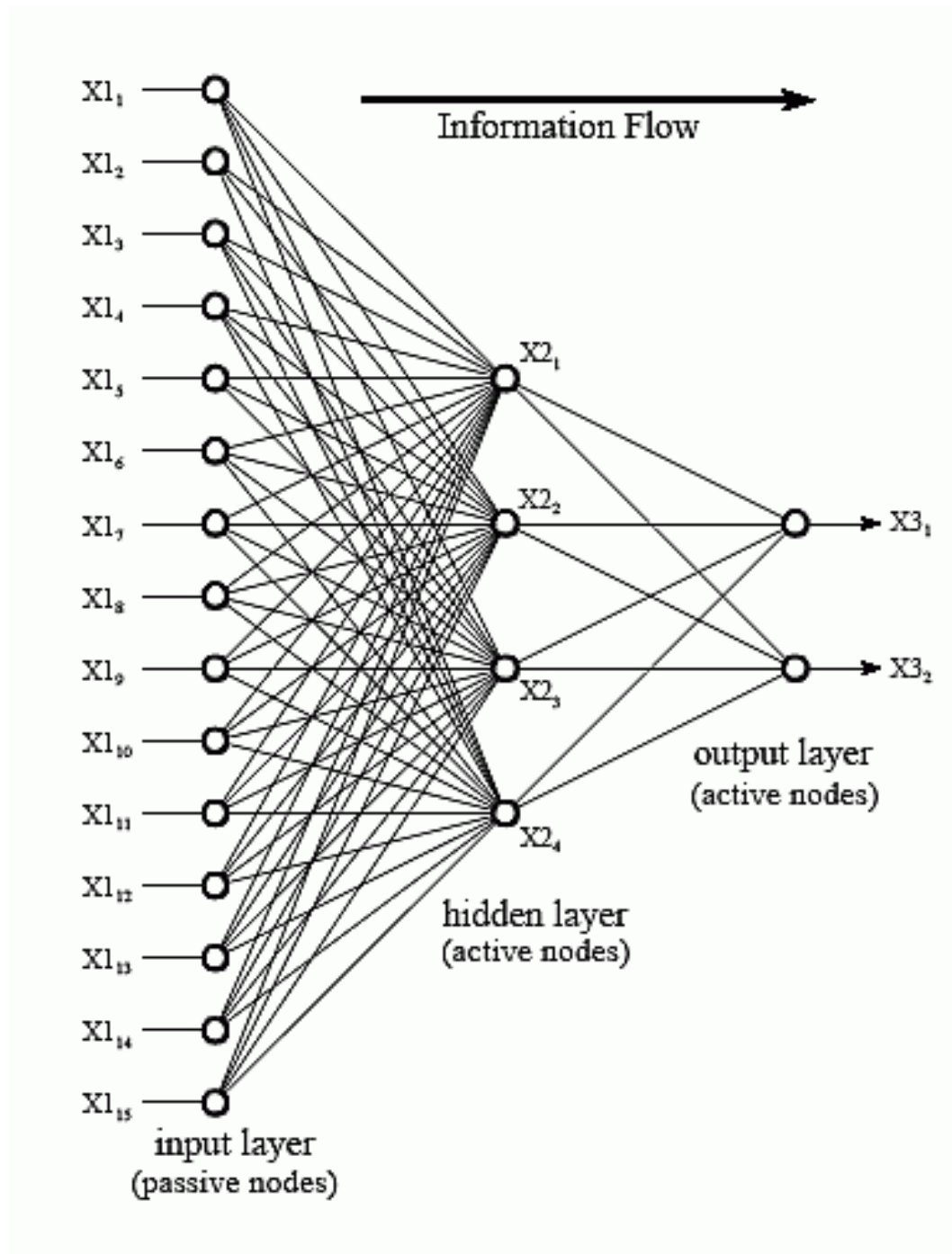
הקדמה לפתרון

תחום הרשתות העצביות הומצא במחצית המאה ה-20, החל במודלים פשוטים של רשתות פרספטרון ורשתות בשכבה אחת. תחום זה מתמקד במודלים חישוביים ביולוגיים המשמשים לפתרון משימות שונות של לימוד מכונה. רשתות עצביות מורכבות משכבות של צמתים מחוברים הנקראים נירונים או פרספטרון (ראה איור 1).



איור 1 - אחת הרשתות נירונים הראשונה שתוכננה.

לכל צומת יש פונקציית אקטיבציה שהפלט שלה הוא הפלט של הצומת, כאשר הקלט של פונקציית האקטיבציה הוא סכום הקלטים המשוקללים. קלט נכנס לרשת באמצעות שכבת הקלט (Input Layer), אשר מקושרת לשכבה אחת של Hidden Layer או יותר שבהן העיבוד מתבצע באמצעות מערכת של קשרים משוקללים. שכבות ה-Hidden Layers מקושרות לשכבת ה-Output Layer שבה פלט הרשת מחושב (איור 2).



איור 2 - מבנה של רשת נוירונים טיפוסית.

ההתפתחות של מדע רשתות עצביות החל בשנת ב-1943 ע"י מאמר שפרסמו שני חוקרים אמריקאים ורן מקלוק וולטר פייטס שהיווה בסיס להתפתחותו של מדע הרשתות העצביות. במאמר הם הציעו מודל פשוט של פעולת הניורון שעליו מבוססות הרשתות המלאכותיות עד היום (Perceptron).

הניורון מקבל קלט (יסומן ב- i), כשלכל אחד מהם משקל יחסי (יסומן ב- w). כל נתון i נשקל על ידי הכפלתו במשקולת המתאימה לו, כאשר התוצאה מסכום הקלטים המשוקלים היא:

$$\langle i, w \rangle = \sum_j i_j w_j = O$$

אם הסכום גבוה מסף ידוע כלשהו הניורון מעביר פלט "1" אחרת "0" - תהליך שמאוד דומה לפעולה של ניורון אמיתי שמפיק אות חשמלי כשהגירוי גבוה מספיק. תפקיד המשקולות הוא לקבוע מהי התבנית שתזוהה וקביעתן היא המשימה המרכזית של תוכנית הרשת העצבית.

כחלק מהשלבים יש לבצע את תהליך ה"אימון" של הרשת כאשר תהליך זה הוא זה שקובע את המשקולות לזיהוי אוטומטי של תבניות ברשת העצבית.

בתהליך האימון, רשת הניורונים מחשבת משקולות עבורם פלט הרשת יהיה דומה לסגמנטציות הנתונות. מציאת משקולות מתאימות יאפשר לקדם את התהליך לסיווג איברים מתוך סריקות CT.

רשת ניורונים היא גישה פורצת דרך ומתקדמת מאוד בתחום הלמידה העמוקה אשר תורמת להתפתחות תחומים רבים כמו זיהוי תווים, זיהוי פנים, זיהוי כתב יד, חיזוי שוק ההון, מערכת זיהוי דיבור, ניתוח טקסט ובסיווג של תמונות כמו בפרויקט זה.

הגישה מחקה את פעולת המוח ומייצרת רשת של ניורונים (Neural Network). כל רשת נבנית על בסיס הניורון הבודד (Perceptron).

רשת מסוג זה מכילה בדרך כלל מספר רב של שכבות המקושרות זו לזו.

הפתרון

הפתרון לבעיה יהיה הכנת המידע בצורה הנוחה ביותר ושימוש בטכניקות למידה עמוקה (Deep Learning), בניית רשת נוירונים (CNN).
הרשת תתאמן על המידע ותלמד להתנהג כמסווג עבור איברים.

מהי המערכת?

פונקציות להכנה ועיבוד של המידע.
רשת נוירונים בעלת שכבות קונבולוציה (CNN) מורכבת ממספר שכבות שונות בגדלים שונים, כגון convolution, maxpool, fully-connected (ראה סעיף 5.2).
בנוסף למערכת תתווסף גם בדיקה שלאחר אימון רשת הנוירונים, אשר דומה לתהליך אימון. על כך יפורט בהמשך.

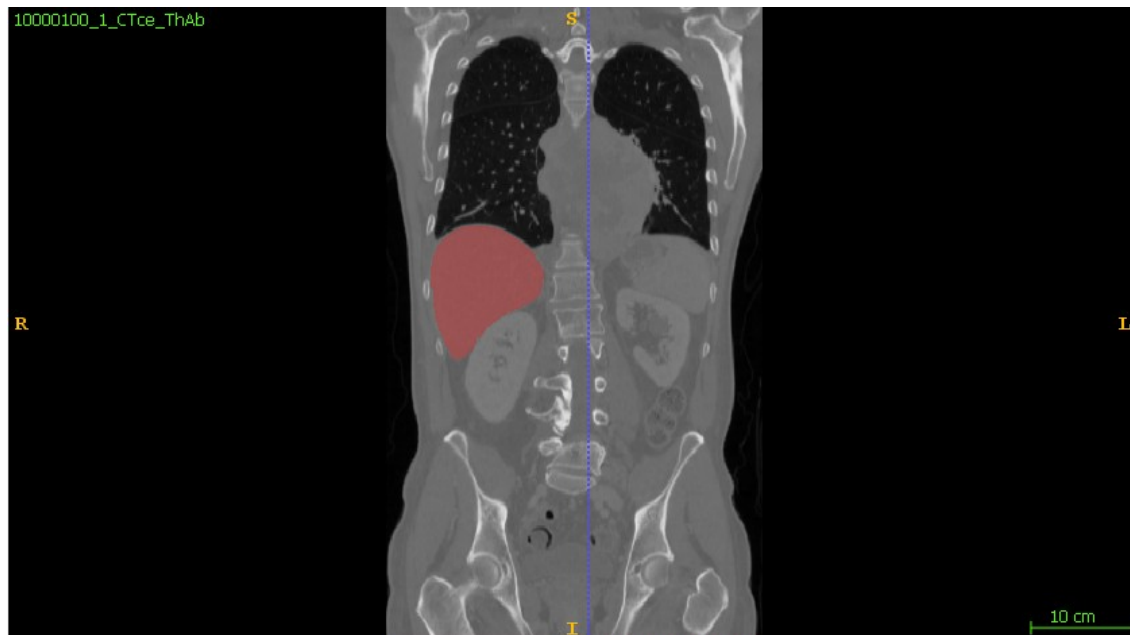
תהליכים ונתוני המערכת

קלט המערכת:

סריקות CT מתוך המאגר "anatomy3-benchmark database" הכולל:

- 20 סריקות CT של בית החזה והבטן בגודל 512X512 עם ניגודיות משופרת, 18 סריקות נועדו לאימון ו-2 סריקות נועדו לאימות.
- 20 סריקות CT של כל הגוף בגודל 512X512 ללא ניגודיות משופרת, 18 סריקות נועדו לאימון ו-2 סריקות נועדו לאימות.

הסריקות עוברות תהליך עיבוד מורכב שבסופו נוצרת רשימה של חלקי תמונה (patches) עבור כל סריקה. הקלט לרשת הנוירונים הוא רשימה של פאצ'ים בגודל 14x14 (יפורט בהרחבה בהמשך).
עבור כל סריקת CT יש בהתאמה סגמנטציה, שנועדה לסמן איברים שונים בסריקת ה-CT (ראה איור 7).
הסיווג של כל איבר בסריקת CT בוצע באופן ידני ע"י מומחה CT. ולכן, זהו מידע איכותי עבור ביצוע אימון ואימות רשתות נוירונים.



איור 3 - סריקת CT ועל גביה סגמנטציה (החלק המסומן באדום).

פלט המערכת:

עבור כל וקטור של פאצ'ים שנשלח לסיווג במערכת, המערכת מוציאה וקטור הסתברויות בגודל של מספר המחלקות. במערכת המתוארת מספר המחלקות הוא 3. המערכת ממירה את וקטור ההסתברויות לוקטור חדש בגודל 3, לאחד מבין הווקטורים הבאים: $[0,0,1]$, $[0,1,0]$, $[1,0,0]$. לאחר מכן המערכת בודקת מהו האינדקס שערכו 1, ומחזירה ערך שלם המייצג את הסיווג של הפאץ' לפי מספר המחלקות. לדוגמא: בהינתן וקטור הסתברויות: $[0.30, 0.25, 0.45]$, הווקטור נהפך להיות $[0,0,1]$ מכיון ש-1 נמצא במיקום 2, יוחזר 2. זאת אומרת נלקח האינדקס שבו ההסתברות הגבוהה ביותר בוקטור.

הסיווגים האפשריים במערכת שפותחה: 0 - לא כבד ולא כלייה, 1 - כבד, 2 - כלייה.

הסבר:

[1,0,0] - לא מייצג כבד וגם לא כלייה.

[0,1,0] - מייצג כבד.

[0,0,1] - מייצג כלייה.

כל אחד מהסיווגים מומרים (באמצעות פונקציית argmax) לערך יחיד כך ש:

0 - [1,0,0]

1 - [0,1,0]

2 - [0,0,1]

כלומר, בהינתן סריקת CT, פלט אפשרי של רשת הניורונים הוא:

[0,0,0,0,1,1,1,2,1,1,2,1,0,0,1,2,1,0]

```
Result:
result = (ndarray) [[[[ 127. 127. 130. ..., 62. 64. 62.] \n [ 119. 121. 134. ..., 67... View
[0:1000] = (list) <class 'list'>: [array([[ 127., 127., 130., 128., 123., 126., 137... View
0000 = (ndarray) [[[[ 127. 127. 130. 128. 123. 126. 137. 145. 136. 107. 71... View
0001 = (ndarray) [[[[ 112. 109. 113. 132. 133. 118. 130. 135. 132. 120. 11... View
0002 = (ndarray) [[[[ 72. 98. 111. 105. 108. 124. 131. 122. 115. 117. 124... View
0003 = (ndarray) [[[[ 125. 125. 129. 125. 126. 127. 133. 137. 131. 115. 83... View
0004 = (ndarray) [[[[ 118. 120. 123. 130. 126. 115. 135. 144. 130. 117. 11... View
0005 = (ndarray) [[[[ 90. 104. 110. 108. 111. 119. 125. 112. 100. 115. 130... View
0006 = (ndarray) [[[[ 124. 133. 141. 126. 118. 121. 128. 144. 140. 129. 96... View
0007 = (ndarray) [[[[ 126. 137. 130. 127. 122. 110. 124. 139. 135. 128. 12... View
0008 = (ndarray) [[[[ 108. 106. 98. 109. 127. 127. 128. 118. 108. 119. 118... View
0009 = (ndarray) [[[[ 63. 76. 97. 110. 113. 110. 108. 117. 122. 121. 126. ... View
0010 = (ndarray) [[[[ 139. 142. 146. 128. 121. 125. 126. 138. 139. 132. 11... View
0011 = (ndarray) [[[[ 122. 126. 120. 128. 127. 105. 108. 119. 119. 130. 13... View
0012 = (ndarray) [[[[ 119. 123. 111. 118. 142. 140. 136. 125. 117. 132. 13... View
0013 = (ndarray) [[[[ 98. 104. 106. 104. 99. 97. 97. 107. 112. 112. 111. ... View
0014 = (ndarray) [[[[ 140. 134. 138. 139. 135. 127. 125. 131. 131. 136. 13... View
0015 = (ndarray) [[[[ 115. 107. 99. 114. 120. 108. 113. 118. 113. 122. 128... View
0016 = (ndarray) [[[[ 111. 118. 119. 116. 130. 134. 128. 124. 124. 142. 14... View
0017 = (ndarray) [[[[ 111. 121. 120. 115. 105. 106. 109. 113. 110. 110. 11... View
0018 = (ndarray) [[[[ 136. 125. 129. 139. 127. 116. 123. 138. 143. 145. 14... View
0019 = (ndarray) [[[[ 114. 110. 102. 104. 96. 101. 116. 117. 114. 119. 125... View
```

```
Result:
result = (ndarray) [[ 0. 1. 0.] \n [ 0. 1. 0.] \n [ 0. 1. 0.] \n ..., \n [ 0. 1. 0.] \n [ 0. 1. ... View
[0:1000] = (list) <class 'list'>: [array([ 0., 1., 0.]), array([ 0., 1., 0.]), array([ 0., ... View
0000 = (ndarray) [ 0. 1. 0.]
0001 = (ndarray) [ 0. 1. 0.]
0002 = (ndarray) [ 0. 1. 0.]
0003 = (ndarray) [ 0. 1. 0.]
0004 = (ndarray) [ 0. 1. 0.]
0005 = (ndarray) [ 0. 1. 0.]
0006 = (ndarray) [ 0. 1. 0.]
0007 = (ndarray) [ 0. 1. 0.]
0008 = (ndarray) [ 0. 1. 0.]
0009 = (ndarray) [ 0. 1. 0.]
0010 = (ndarray) [ 0. 1. 0.]
0011 = (ndarray) [ 0. 1. 0.]
0012 = (ndarray) [ 0. 1. 0.]
0013 = (ndarray) [ 0. 1. 0.]
0014 = (ndarray) [ 0. 1. 0.]
0015 = (ndarray) [ 0. 1. 0.]
0016 = (ndarray) [ 0. 1. 0.]
0017 = (ndarray) [ 0. 1. 0.]
0018 = (ndarray) [ 0. 1. 0.]
0019 = (ndarray) [ 0. 1. 0.]
```

איור 4 - משמאל: ניתן לראות את וקטור הפאצ'ים בגודל מוגדר 1000 (גודל Batch) מימין: ניתן לראות את הסיווג האמיתי של כל פאצ' על בסיס סגמנטצית המקור בהתאמה לפני השימוש בפונקציית Argmax .

4. תיאור הפתרון המוצע

הפתרון הינו בניית רשת נוירונים בעלת שכבות קונבולוציה (CNN) המשמשת לעיבוד תמונה, בפרט עבור תמונות רפואיות, במטרה לסווג את הפאצ'ים לפי האיבר שמופיע באותו פאץ'.

המטרה היא לבנות רשת נוירונים, לאמן אותה עד לקבלת אחוזי דיוק גבוהים, לסווג את הפאצ'ים מתוך וקטור הפאצ'ים ולאחר מכן לסווג איברים מתוך סריקת CT. האתגר יהיה למצוא את כלל הפרמטרים עבורם הרשת תשיג תוצאות טובות. הוגדר ברשימת הדרישות כי תוצאה טובה היא דיוק של מעל 90%.
כמו כן, הפתרון דורש שלב מקדים של עיבוד המידע הנכנס לרשת הנוירונים, עליו נפרט בהמשך.

תיאור הכלים המשמשים לפתרון

- numpy - מחלקה המיועדת לחישובים מדעיים, ובפרט עבור חישוב יעיל של מטריצות. ממומשת באמצעות Python.
- TensorFlow – מחלקה שבאמצעותה ניתן לממש רשתות נוירונים ואלגוריתמים ב-Deep Learning. ממומשת באמצעות Python.
- TensorBoard - כלי לטעינת לוגים שנוצרו על ידי TensorFlow. יוצר גרפים וויזואליזציה של הרשת.
- ITK-Snap - תוכנה לפתיחת סריקות CT וסגמנטציות (קבצי NIFTI).
- Pycharm - עורך הקוד בפרויקט זה.
- AWS - שירותי הענן של Amazon לצורך הקמת מחשב ענן המאפשר אימון רשתות נוירונים.

הקדמה כללית למימוש הפתרון


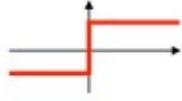
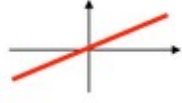
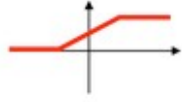


ההקדמה נועדה כדי לתאר את הפונקציות והאלגוריתמים שעליהם התבססה רשת הנירונים.

רשתות קונבולוציה עצביות

כיום, אחד הענפים המתקדמים ביותר של רשתות עצביות הוא למעשה סיווג תמונות. שכבת הקלט של הרשת היא תמונה או חלק של תמונה, והפלט הוא וקטור ההסתברויות לפי מספר המחלקות המסווגות.

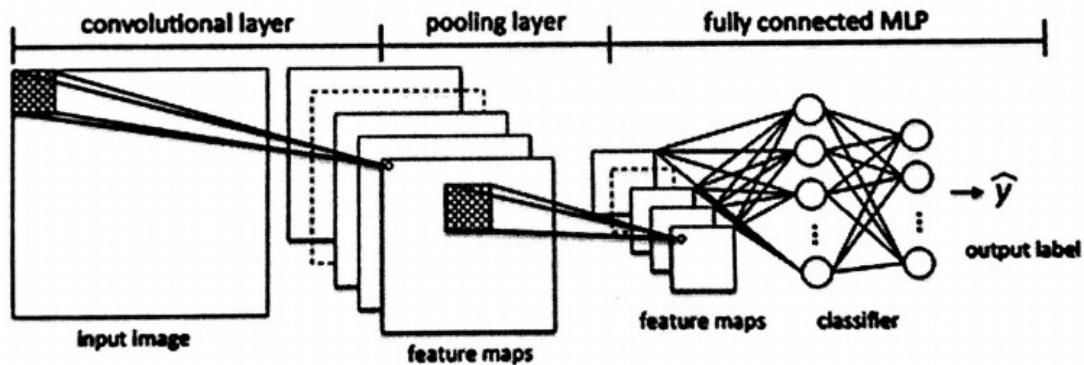
רשתות קונבולוציה עם שכבת קונבולוציה (ConvNets) דומות מאוד לרשתות עצביות בסיסיות (ANN), שתיהן מורכבות מצמתים בעלי משקולות למידה ובעלי פונקציות הפעלה (Activation Function) (ראה איור 5). המשקולות מתוקנות על סמך פונקציית הפסד (Loss Function) והפלט הוא וקטור של הסתברויות.

ההבדל בין רשתות עצביות ורשתות קונבולוציה הוא שרשתות קונבולוציה מניחות כי הקלט של הרשת הוא תמונה ומוסיף תכונות נוספות על מנת לנתח את מאפייני התמונה.

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer NN	

איור 5 - סוגים שונים של פונקציות אקטיבציה.

הארכיטקטורה של רשתות קונבולוציה עצביות (איור 6) מורכבת מחמש אבני הבניין המסודרות בשכבות. לכל אחת מהשכבה יש מבנה שונה ושימוש שונה.



איור 6 - ארכיטקטורה בסיסית של רשתות קונבולוציה.

Input Layer - זוהי השכבה הראשונה ברשת הניורונים. הקלט לשכבה היא תמונה או חלק מתמונה. כמו כן התמונה מורכבת מפיקסלים המיוצגים על ידי ערך מספרי, ולכן ניתן לבצע חישובים על התמונה.

מכיוון שלרוב תמונה ניתנת להקטנה למימדים קטנים יחסית מבלי לפגוע כמעט באיכות במידע, ניתן להכניסה ישירות כקלט לרשת הניורונים. עם זאת, בהתעסקות עם תמונות CT תלת מימדיות, לא רצוי להקטין את התמונה עד כדי כך, כדי לא לאבד פרטים וכדי לאפשר לרשת לעבוד עם המידע המקורי ככל שניתן. לכן, בדרך כלל נהוג לחלק תמונות גדולות לבלוקים (Patches).

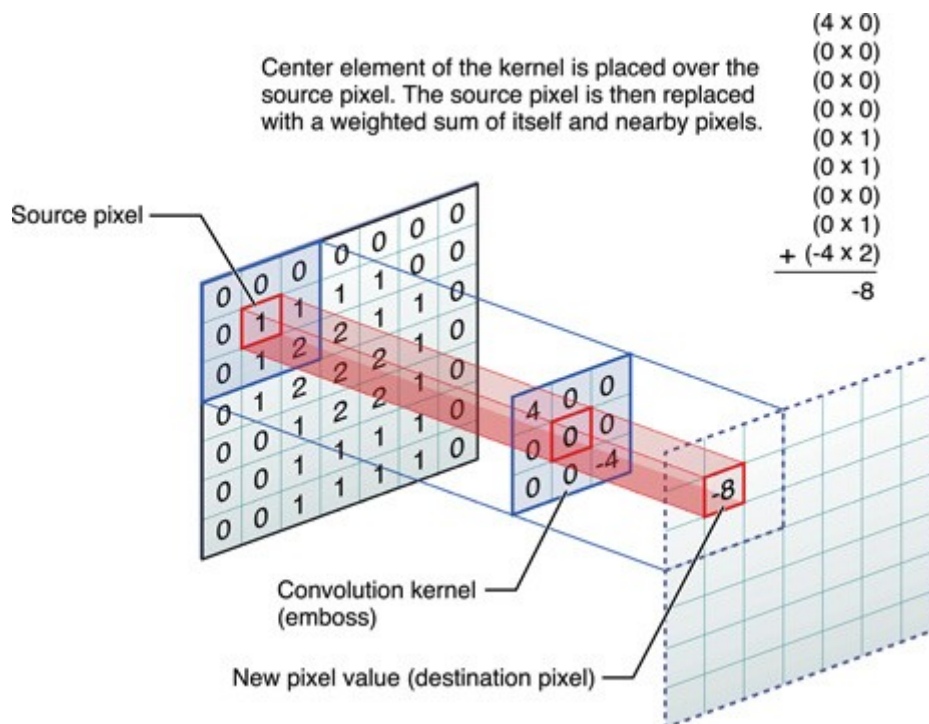
Convolution Layer - שכבה זו היא הליבה של רשת ניורונים קונבולוציה (CNN), המטרה העיקרית של שכבה זו היא לחלץ פיצ'רים משכבת הקלט ולייצר מאפיינים (Features).

השגת המאפיינים מתבצעת ע"י אופרטור קונבולוציה על התמונה. גרעין הקונבולוציה (פילטר) יכול להיות דו מימדי או תלת מימדי.

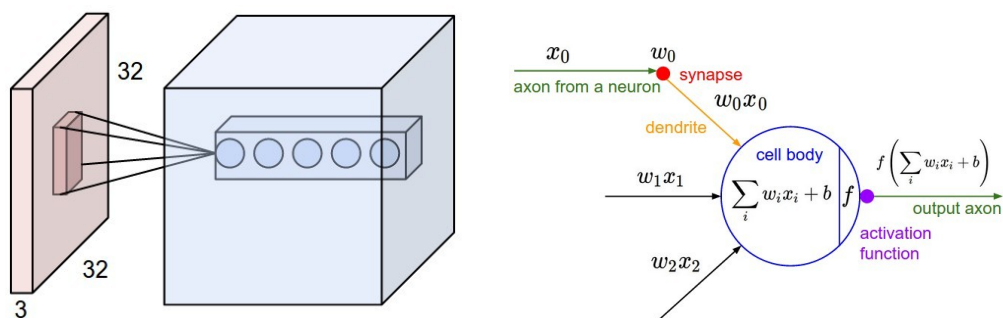
הפלט שמתקבל מהפעולה מועבר ל-Activation Function.

גודל הפלט של שכבה זו יכול להשתנות בשל גודל שונה בין הצעדים (strides).

בדרך כלל הגודל יהיה $[X*Y*N]$ כאשר N זה מספר ה-Kernel Filter (ראה איור 7).

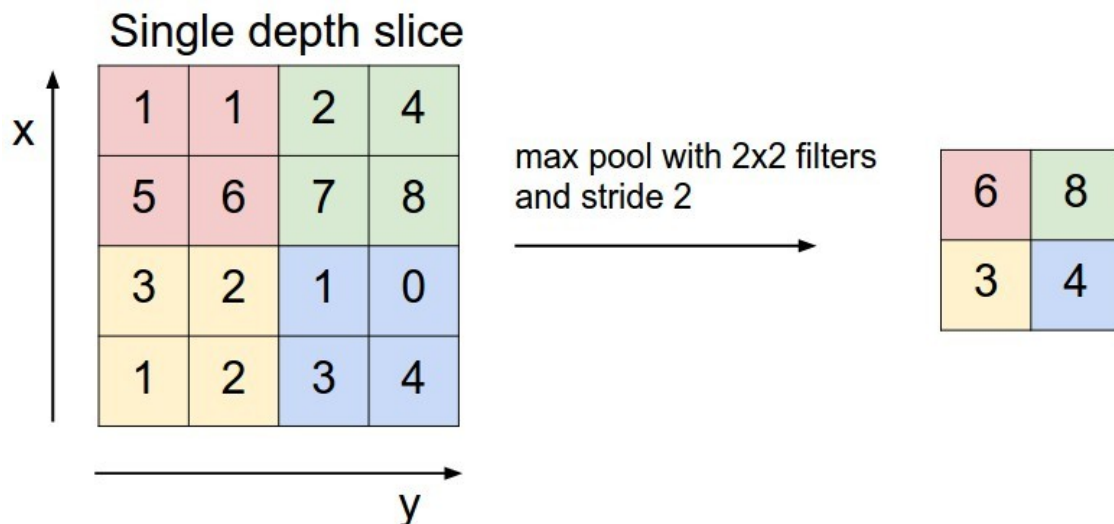


איור 7 - מתאר את הפעולה של שכבת הקונבולוציה.



איור 8 - משמאל קונבולוציה, מימין סכימה של הפלטים מהשכבה הקודמת והכנסתם לפונקציית אקטיבציה.

Max Pool Layer - שכבה זו מבצעת דגימה (Downsampling) לאורך הממדים המרחביים של התמונה, וכתוצאה מכך נקבל פלט בגודל קטן יותר. בדרך זו נבחרים רק המאפיינים המשמעותיים של התמונה עבור השכבה הבאה. גודל הפלט של שכבה זו תלוי בפרמטר של שכבת ה-Max Pool. למשל, עבור גודל דגימה בגודל 2, גודל הפלט של שכבה זו יהיה: $[X/2 * Y/2 * N]$, כאשר N משמעותו המימד השלישי (ראה איור 8). בדרך כלל, מספיק לשמור ערכים משמעותיים מתוך מטריצה, כך שלא נפגע באיכות המידע. לכן ניתן לבצע Max Pool בין השכבות השונות כדי להקטין את כמות המידע עליה עובדים ולהשאיר את המידע המשמעותי.



איור 9 - הקטנה של מטריצה על ידי **Max Pool**. תמונה בגודל $4 \times 4 \times 1$ ופילטר מקסימום בגודל 2×2 מקטין את התמונה פי 2.

Fully Connected Layer - שכבה זו פועלת כמו שכבת רשת נוירונים רגילה. השכבה מחברת את כל הקלטים מהשכבה הקודמת אל כל M הנוירונים המרכיבים אותו. למעשה זהו שלב איסוף כל הנתונים שהצטברו מהחישובים בשכבת הקונבולוציה והפעלת פונקציית אקטיבציה על הערך שהתקבל. גודל הפלט משכבה זו הוא [M].

Output Layer - שכבה זו מחשבת את ההסתברות שנקבעה עבור כל מחלקה מתמונת הקלט. השכבה מורכבת מ-K נוירונים כאשר כל אחד מהם מחובר לכל הפלטים של השכבה הקודמת. גודל הפלט משכבה זו הוא [K], שזה למעשה מספר המחלקות.

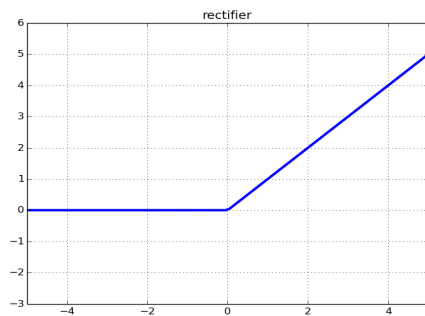
CNN Parameters

- Activation Functions

כל צומת (נירן) סוכם את מכפלת כל הקלטים עם המשקולות ומעביר את התוצאה כקלט לפונקציית ה Activation Function.

יש סוגים שונים של Activation Functions בשימוש תחום רשתות הנירונים כמו למשל Sigmoid, Tanh ו-Relu. בפרויקט זה נעשה שימוש ב-Relu.

הפונקציה Relu לא רוויה כמו sigmoid או tanh, בשל כך מובילה להתכנסות יותר מהירה. זוהי הפונקציה הפופולרית היום בשימוש ברשתות ניורונים.



$$f(x) = \max(0, x)$$

Output Format -

גם לשכבה האחרונה של הרשת יש Activation Function שתפקידה בשכבה זו היא לחשב את הסיווג.

בפרויקט זה נעשה שימוש בפונקציית Softmax.

הפונקציה Softmax מנרמלת את פלט הנוירון לווקטור ערכים (הסתברויות) בטווח [0,1]. הסכום של ווקטור הפלט הוא 1.

האינדקס בווקטור בעל הערך הגבוה ביותר הוא הסיווג שנבחר.

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}}$$

Loss Function

במהלך שלב האימון, הרשת מחשבת את ההפסד, כלומר מתבצע חישוב של הסיווג האמתי מול הסיווג שקיבלנו מהרשת (החיזוי).

בפרויקט זה נעשה שימוש ב-Categorical Cross-Entropy.

הפונקציה Categorical Cross-Entropy פועלת באותו עקרון כמו Softmax Function, כלומר הערך של פונקציית ההפסד מחושב ע"י הנוסחה:

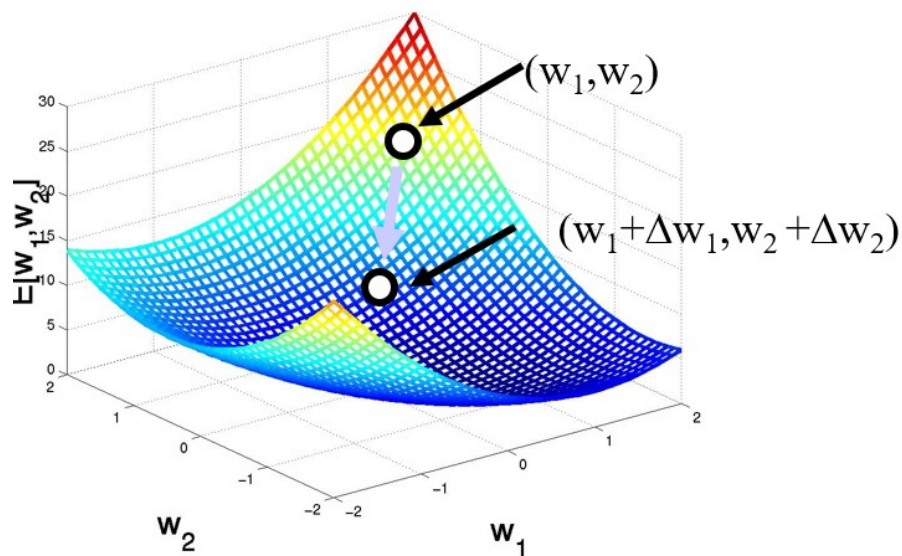
$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right).$$

- Parameter Update Method

לאחר שקובעים את מספר שכבות הרשת, יש צורך לקבוע את ערכי המשקולות. משטח הטעות - error surface - כל אחת מהמשקולות מהוות מימד במרחב, המימד הנוסף יהיה טעות הרשת. כל קונפיגורציית משקולות אפשרית ניתנת לייצוג כמשטח במרחב זה.

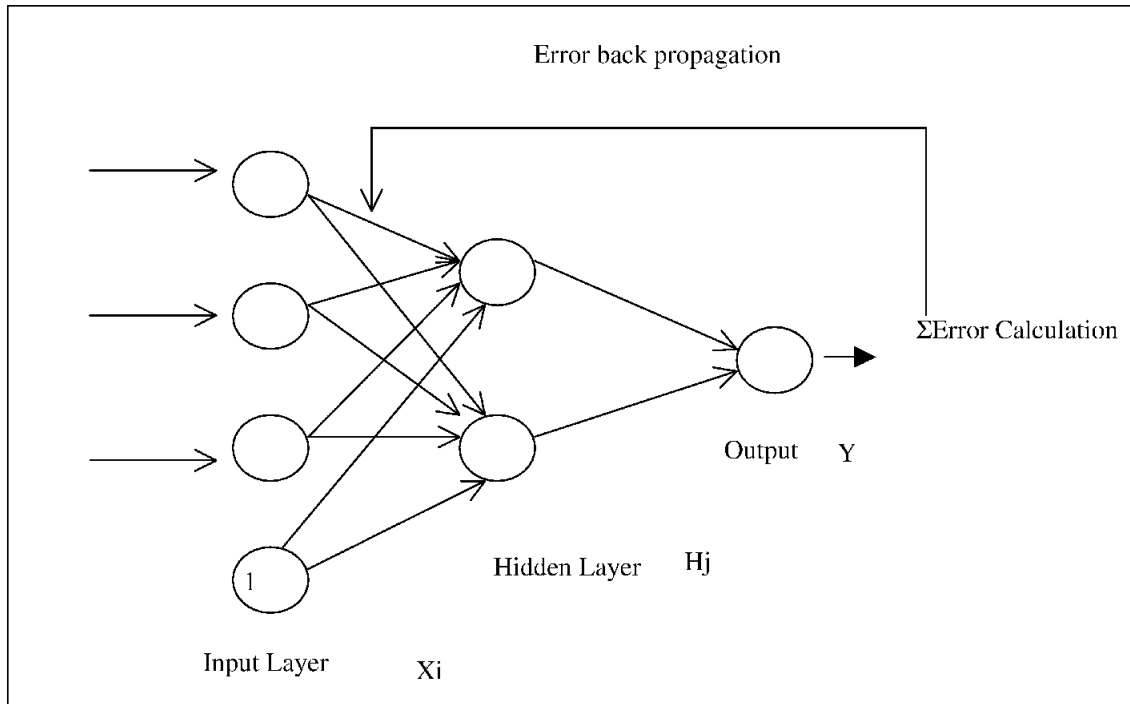
הגרדיאנט היינו ווקטור של הנגזרות החלקיות, הגרדיאנט מחושב במהלך פעולת Backpropagation כדי לעדכן את המשקולות של הרשת.

מטרת תהליך הלימוד - למצוא את נקודת המינימום במשטח זה ובכך למצוא את השגיאה המינימאלית.



איור 10 - עדכון משקולות

רשת הנוירונים בעלת שכבות קונבולוציה (CNN) מקבלת תמונה או חלקים מתמונה ומבצעת תהליך אימון המורכב מחישוב הפלט (Feedforward) ולאחר מכן תיקון המשקולות (Backpropagation) (ראה איור 3).



איור 11 - תיאור שלב ה-Backpropagation.

ישנן מספר שיטות כדי לעדכן את המשקולות, למשל Gradient Descent או Momentum. בפרויקט זה נעשה שימוש ב-Nesterov Momentum עבור עדכון המשקולות.

Nesterov Momentum:

μ נקרא מקדם המומנטום.

בשיטה זו, ראשית מעדכנים את וקטור המשקולות בעזרת Momentum מהאיטרציה הקודמת - זוהי הערכה משוערת לגבי הערכים הנכונים יותר בווקטור המשקולות. לאחר מכן הגרדיאנט והוקטור משקל מתעדכן בהתאם. נציין כי Nesterov Momentum מבטיח שיעור התכנסות טובים יותר.

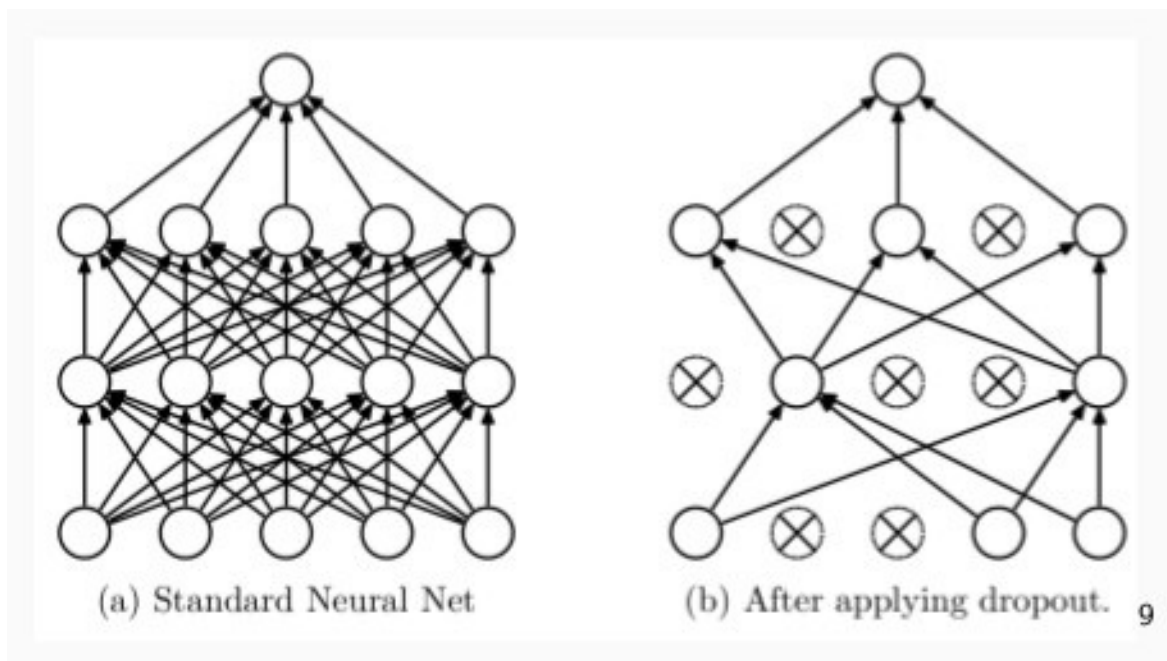
$$\hat{W} = W + \mu \cdot V$$

calculate Gradient using \hat{W}

$$V = \mu \cdot V - LearningRate \cdot gradient$$
$$W = W + V$$

Dropout

בשיטה זו מבטלים מספר נוירונים ברשת באופן אקראי לפי פרמטר בטווח $[0,1]$. פעולה זו נועדה בכדי למנוע מנוירונים להיות במצב של co-adapting גבוה ולמנוע מצב של Overfitting על המידע של האימון. בזמן הלמידה, בכל איטרציה על הרשת נאפס חלק מהנוירונים כך שתתקבל תת רשת אקראית, ממשיכים באיפוס נוירונים עד למצב שעברנו על כל האפשרויות של הרשתות האקראיות (ראה איור 6).



איור 12 - ביטול של צמתים ברשת.

תיאור המערכת שמומשה

פותרת מערכת הכוללת בתוכה 3 חלקים עיקריים:

1. פונקציות לעיבוד המידע והמרתו לוקטור של פאצ'ים - המידע בצורתו הטבעית היינו סריקות CT (קבצי NIFTI). כדי שרשת הנוירונים תוכל להתאמן על סמך מידע זה, יש צורך להמירו למבנה נתונים המאפשר לבצע מניפולציות על התמונה (למשל הקטנה או הגדלה של תמונה, חיתוך התמונה, הזזות וכדומה). את התמונות ניתן לייצג בקלות בעזרת מטריצות, ובפרט בעזרת שימוש במחלקה numpy, המאפשרת התעסקות עם תמונות. עם זאת, המחלקה numpy אינה מסוגלת להתעסק עם קבצים מסוג NIFTI ולכן נעשה שימוש במחלקה SimpleITK על מנת לטעון סריקות ולהמירן ל-numpy array. לאחר טעינה של סריקה אחת, מתקבלת מטריצה תלת מימדית בגודל של כ- 512X512X413 פיקסלים, מטריצה זו מחזיקה את ערכי הפיקסלים בהתאמה עם הסריקה המקורית (קובץ ה NIFTI). על מנת לייעל את זמן הריצה של אימון הרשת, ניתן להתעלם מחלקים לא רלוונטים בתמונה, כלומר ניתן להתמקד רק באזור שבו נמצאים האיברים שנרצה לסווג. כמו כן, ניתן להקטין את מימדי התמונה פי 2 (Downsampling), באופן שלא יפגע במידע. בנוסף, כדי לייצר מידע מגוון יותר עבור רשת הנוירונים (ולהימנע ממצב של Overfitting), ישנה פונקציה המבצעת הזזה של הפאצ'ים בוקטור על מנת לשנות את הסדר שלהם וליצור, לכאורה, תמונה שונה. על ידי כך ניתן לייצר מידע נוסף ומגוון יותר עבור הרשת. כל סריקה חולקה לפאצ'ים (Patches) בגודל 14x14 כדי להקטין את כמות המידע הנכנסת לרשת ולאפשר הרצה עם mini batches. כדי לבצע חלוקה של שלמה של הפאצ'ים - יצרנו מטריצה חדשה בגודל 518X518X413. הרציונל של mini batches הוא להכניס כמויות קטנות של מידע לרשת הנוירונים כדי שהלמידה תהיה נקודתית יותר. כמו כן mini batches מאפשר הרצה על מערכת מוגבלת משאבים, מכיוון שלא כל המידע נטען בפעם אחת.
2. רשת נוירונים בעלת שכבות קונבולוציה מבוססת על הספריה TensorFlow - מומשה רשת נוירונים בעלת שכבת קונבולוציה אחת (Simple Net). מאילוצי משאבים, לא מומשו רשתות נוירונים עמוקות יותר (בעלות מספר רב של

שכבות קונבולוציה).

המחשב העיקרי שבו השתמשנו עבור האימון:

Intel i7 3960X Extreme - 12 Cores

AMD Radeon R7 200

8GB DDR3 1600MHz RAM

240GB SSD

מחשב נוסף:

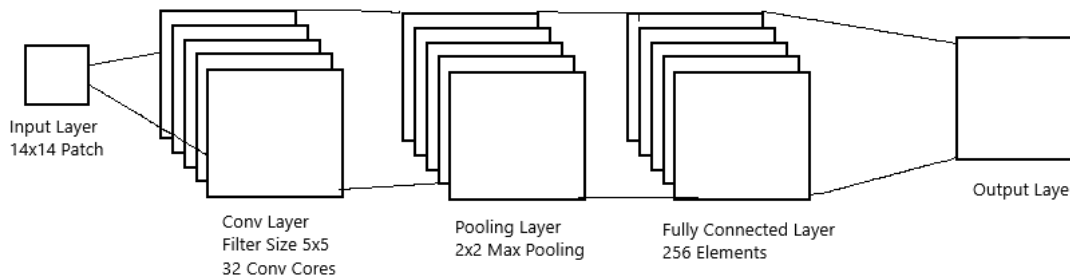
Intel Core i5-4210

GB RAM 4

3. פונקציה לבדיקת המשקולות שהתקבלו מרשת הנוירונים - הפונקציה דומה מאוד לתהליך של אימון, פרט לכך שהמשקולות אינן עוברות תיקון. כלומר, ישנה רשת נוירונים המקבלת כקלט וקטור של פאצ'ים, ומוציאה כפלט את וקטור הסיווג של כל אותם פאצ'ים. הדגש היחידי לשלב הזה הוא שימוש בסריקת CT שלא הייתה בשימוש בתהליך האימון, וזאת על מנת לוודא שהרשת אינה "מכירה" את הסריקה.

תוצאות הרשת

הרשת שפיתחנו באמצעות TensorFlow בעלת שכבת קונבולוציה אחת (Simple Net):



איור 13 - רשת הניורונים Simple Net.

כדי לבדוק את תקינות הרשת, תחילה עבדנו עם כמות מידע קטנה (מספר פאצ'ים בודדים). רק לאחר מכן, כאשר וידאנו שהרשת אכן מסווגת פאצ'ים בודדים, הכנסנו יותר מידע (סריקה שלמה, מספר סריקות).

תוצאות הרשת מתחלקות למספר חלקים כתלות במידע:

1. שימוש ב-3 פאצ'ים בודדים מתוך סריקת CT אחת וסיווג (החלק המתאים מהסגמנטציה). שלב זה נועד כדי לבדוק האם הרשת עובדת כנדרש. הציפייה היא לקבל מצב של Overfitting בצורה יחסית מהירה, מכיוון שכמות המידע קטנה מאוד. כמו כן, בהנחה שהרשת עובדת, הדיוק אמור לשאוף ל-100%.

ואכן, לאחר אימון קצר, דיוק הרשת היה 100%. כמו כן, השוואה ידנית של הסיווג שהתקבל עבור כל פאצ' מהרשת אל מול הסיווג המקורי, הראה שהמסווג עבד כנדרש. תהליך זה אימת שהרשת אכן מתפקדת כנדרש.

2. שימוש בסריקה CT אחת בלבד ו-2 סגמנטציות של כבד וכליה. גם שלב זה נועד כדי לבדוק האם הרשת עובדת כנדרש.

בדומה לסעיף הקודם, המטרה היא לוודא שהרשת מסווגת את כלל הפאצ'ים מתוך סריקה אחת באופן מדויק. לאחר הרצת אימון של מספר שעות, הרשת

הגיעה לאחוזי דיוק גבוהים יחסית של כ-90%. השוואה אקראית של סיווגי הפאצ'ים אל מול הסיווג האמיתי מראה שהרשת מסווגת נכון.

3. שימוש ב-3 סריקות CT (אחד נלקח לאימות), ו-2 סגמנטציות של כבד וכליה

4. שימוש ב-10 סריקות CT (שניים נלקחו עבור אימות), ו-2 סגמנטציות של כבד וכליה.

על בסיס המחקר עליו מסתמך הפרויקט, החלק הראשון של התוצאות בא לשחזר את המחקר, ע"י שימוש ב-TensorFlow. הפרמטרים היו זהים לאלה של המחקר (גודל הרשת, קצב לימוד, פונקציות האקטיבציה וההפסד וכדומה).

5. שימוש ב-10 סריקות CT (שניים נלקחו עבור אימות) ו-4 סגמנטציות של כבד, כליה, אבי העורקים וטחול. בשלב זה הוקטן קצב הלימוד ל-0.00001, על מנת לאפשר תיקון משקולות סולידי יותר.

ערכי הפרמטרים שבאו לידי שימוש ברוב המקרים מפורטים כדלהלן (מוסברים במילון המונחים):

Simple Net Network

Segmentation Ratio = 0.75

stddev = 0.1

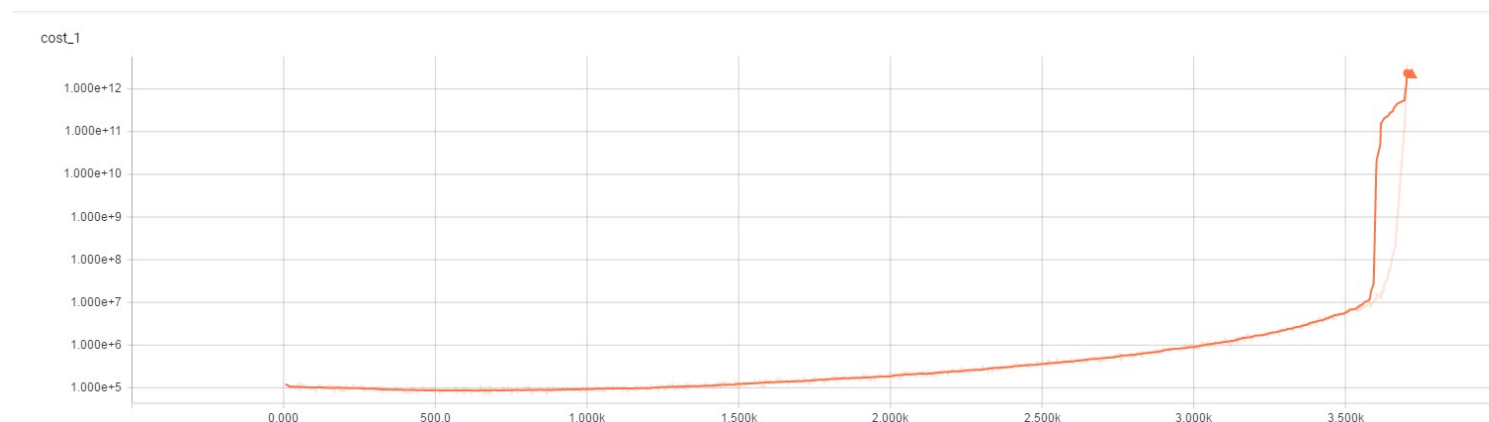
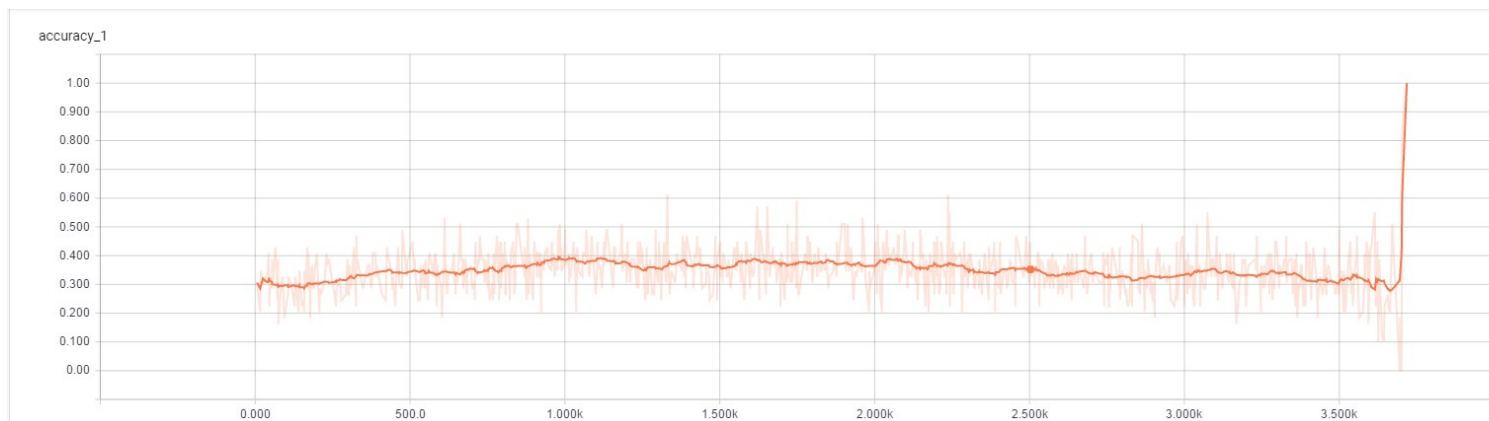
Learning Rate = 0.0001

$\mu = 0.9$ (for Momentum)

Batch Size = 1000

Billion+ Iterations (about 2 weeks of training) 1

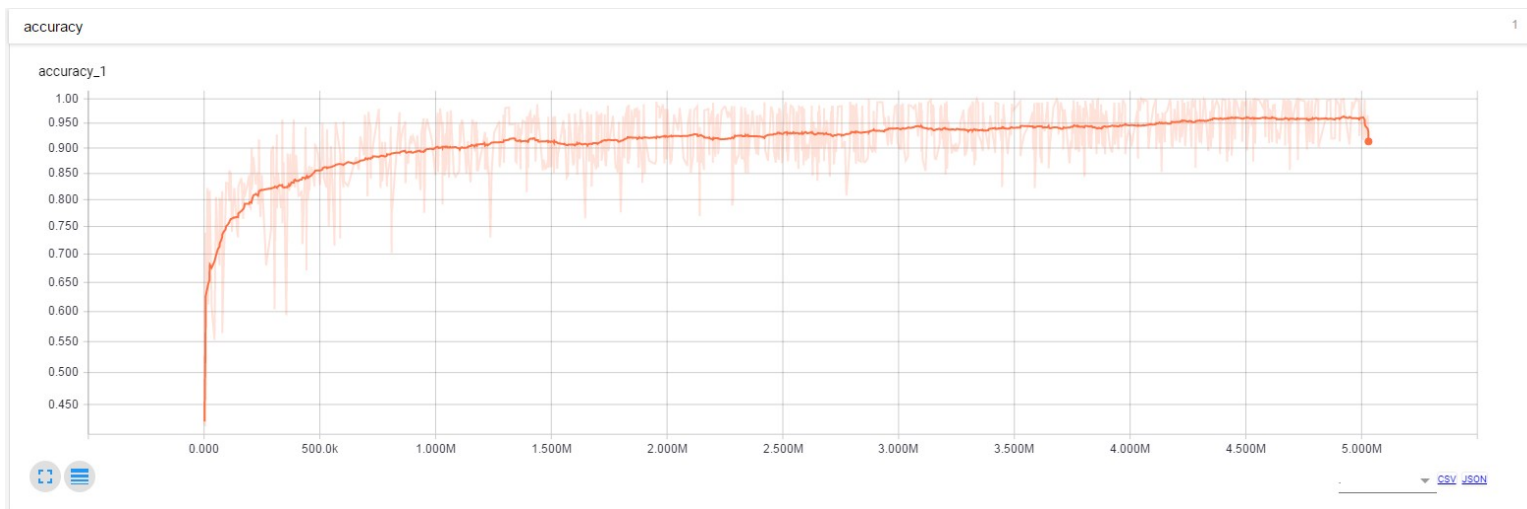
באימונים הראשונים של הרשת התקבלו תוצאות לא טובות, שנבעו ממספר באגים בקוד.
הדבר גרם לתהליך אימון כושל ודיוק נמוך מאוד שעמד על כ-35% בממוצע, עבור 3
פאצ'ים בודדים. (ראה איורים 14 ו-15).



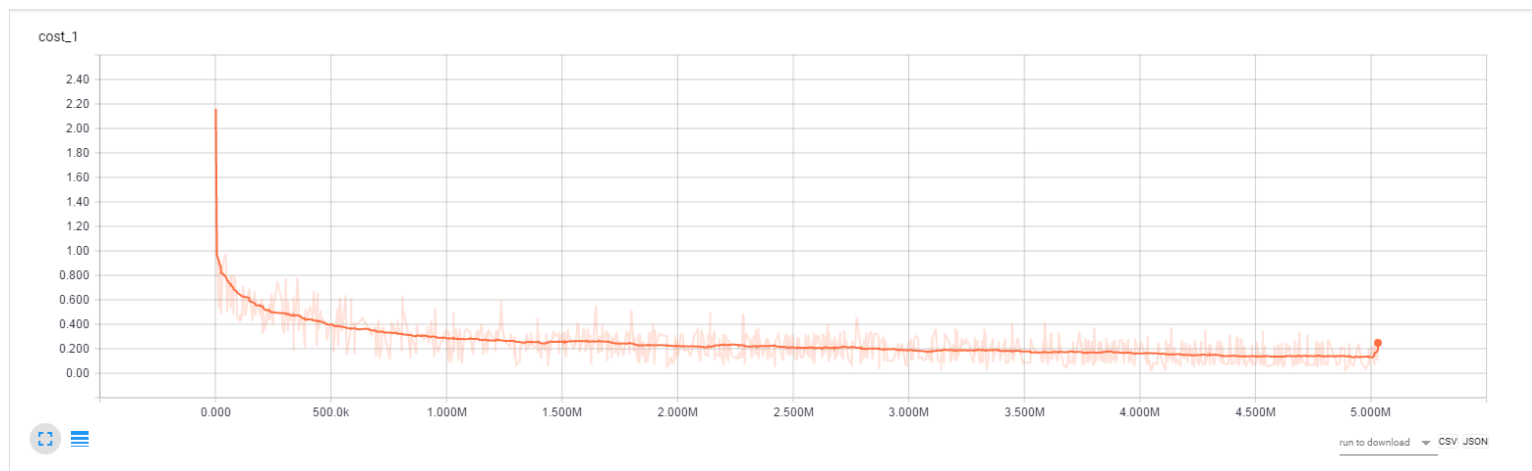
איור 14 - גרף הדיוק (Accuracy) של רשת Simple Net בתחילת הפרויקט.

איור 15 - גרף ההפסד (Loss) של רשת Simple Net בתחילת הפרויקט.

לאחר תיקון הבאגים, הרצנו אימון נוסף על הרשת. הרצה על 10 סריקות (2 לאימות) וכשבועיים רצופים של אימון, דיוק הרשת הגיע לכ-95% וההפסד היה מינימלי (ראה איורים 16 ו-17).

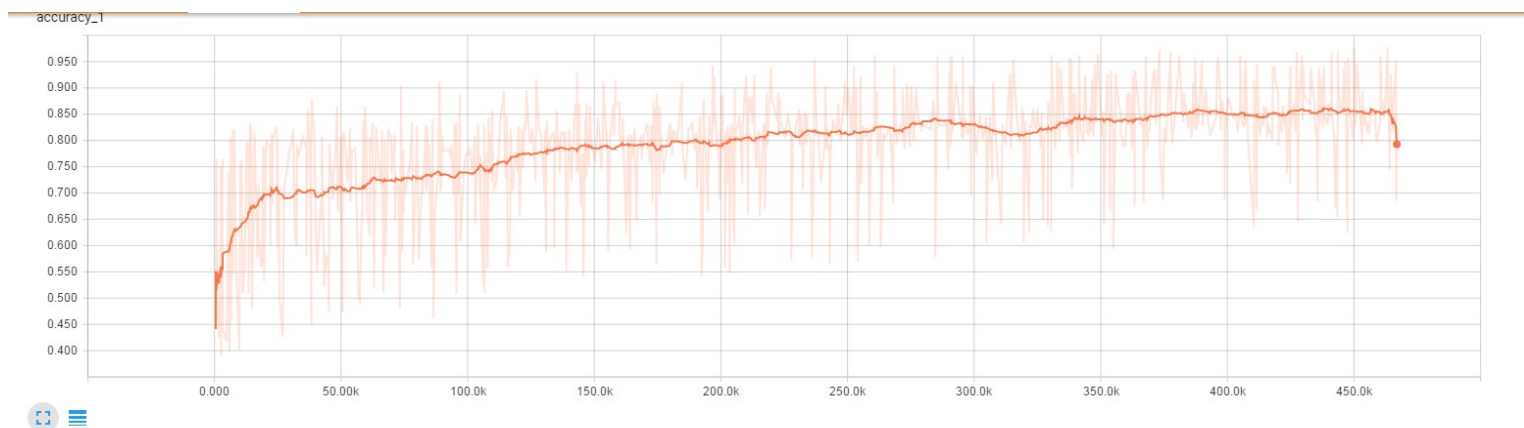


איור 16 - גרף הדיוק (Accuracy) של רשת Simple Net.

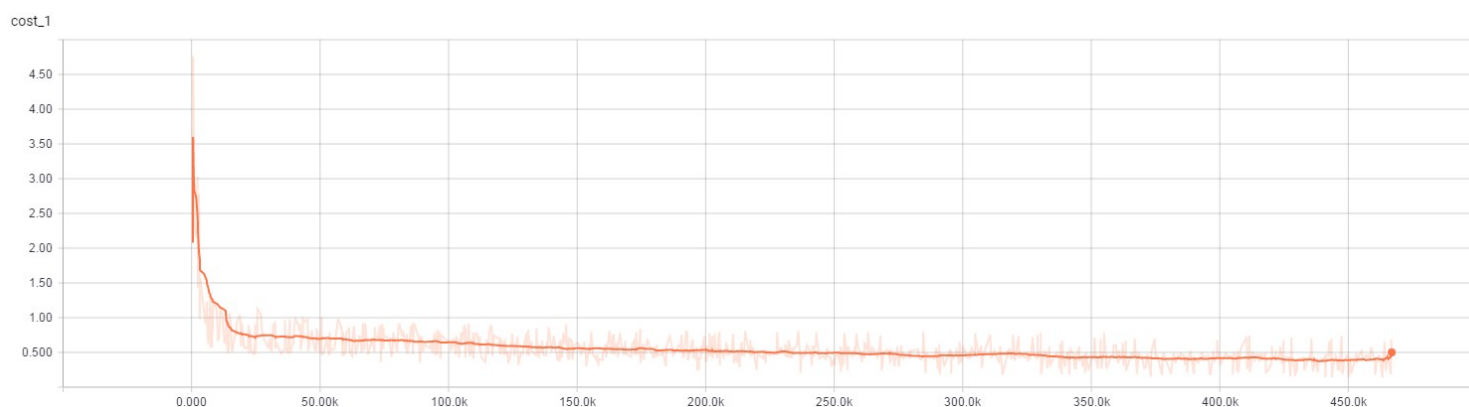


איור 17 - גרף ההפסד (Loss) של רשת Simple Net.

2. שימוש ב-10 סריקות CT (שניים נלקחו עבור אימות), ו-4 סיגמנטציות של כבד, כליה, אבי העורקים וטחול. הפרמטרים דומים פרט לקצב הלימוד ששונה ל-0.00001. כמו כן, את האימון במקרה זה הרצנו כשבוע (כ-500 מליון איטרציות).



איור 18 - גרף הדיוק (Accuracy) של רשת Simple Net עבור 4 סיגמנטציות.



איור 19 - גרף ההפסד (Loss) של רשת Simple Net עבור 4 סיגמנטציות.

התוצאות שהתקבלו עבור שני המקרים האחרונים טובות מאוד. התוצאות מראות כי הרשת מצאה משקולות אשר עבורן פלט הרשת דומה מאוד לסיווג האמיתי של הפאצ'ים, כ-5% שגיאה במקרה הראשון וכ-15% שגיאה במקרה השני. הפלט של הרשת בשני המקרים היה סיווגי הפאצ'ים, שאת סיווגים אלו היה צריך להמיר בסיגמנטציה. עם זאת, השימוש ב-TensorFlow יחד עם SimpleITK עבור הוצאה של סיגמנטציה לא היה טבעי ואף מסובך, ולכן, מפאת חוסר בזמן, בפרויקט זה אין סיגמנטציות להמחשת הצלחת אימון הרשת.

בהשוואה למחלקות אחרות (למשל DeepMedic) אשר מתאימות יותר לאימון רשתות עבור ניתוח תמונות רפואיות והוצאת סיגמנטציות, TensorFlow היא מחלקה במיועדת ל-Deep Learning ו-Machine Learning. ואת ההתאמות עבור הוצאת הסיגמנטציה היה צריך לממש באופן עצמאי, או ע"י שימוש במחלקה אשר האינטגרציה בינה לבין TensorFlow היא אפשרית ונוחה.

בפרויקט זה, לא הוערך נכונה הקושי הנ"ל ולכן לא נמצא פתרון מעשי במסגרת הזמנים של הפרויקט. מחלקות כגון SimpleITK (שהייתה בשימוש בפרויקט זה), NiBabel וכן Nilearn מציעות ייצוא של קבצי NIFTI ובפרט סגמנטציות, אך הקושי עצמו היה בבניית הסיגמנטציה ולא בייצוא שלה. למחלקה SimpleITK היה תיעוד בסיסי למדי, מה שהקשה מאוד בשלב ייצוא הסיגמנטציה. במחקר שעליו מתבסס פרויקט זה נעשה שימוש במחלקה SimpleITK לכן כדי לנסות לשחזר את המחקר בצורה המדויקת ביותר, גם בפרויקט הזה נעשה שימוש ב-SimpleITK. ייתכן כי החלטה זו הייתה שגויה והיה צורך לבדוק גם מחלקות אחרות טרם תחילת הפרויקט.

מצד שני, הצלחת הפרויקט הזה עשויה להימדד על סמך תהליך אימון מוצלח וייצוא המשקולות. המשקולות יכולות לבוא לידי שימוש בכל פרויקט אחר ולמעשה להמשיך את תהליך סיווג האיברים וייצוא סגמנטציות בסופו של התהליך. הפרויקט מכיל תשתית הניתנת להרחבה בקלות אשר בנויה ב-TensorFlow. ניתן לשנות באופן פשוט כל מרכיב במערכת כמו למשל שינוי פרמטרים של שכבה אחת, שינוי מבנה של רשת נוירונים, שינוי אופן אתחול המשקולות. לכן, במידה ורוצים להרחיב את פרויקט זה כדי לסווג איברים נוספים, ניתן לעשות את הדבר בקלות ומבלי לכתוב הכל מחדש.

5. תוכנית בדיקות

כחלק מתהליך הפיתוח של רשת הנורונים, עלינו לבצע בדיקות כדי לוודא את תקינות המערכת.

בכל שינוי של פונקציה נרצה לבצע בדיקה, זאת כי כל שלב בתהליך עשוי להשפיע על הביצועים (רמת הדיוק של רשת הנורונים).

ניתן לחלק את הפרויקט לשני חלקים עיקריים: מידע ורשת נורונים.

בחלק של המידע עלינו לטעון את כל התמונות ולבצע נרמול של המידע (כלומר למחוק רעש ומידע לא רלוונטי).

בחלק של רשת הנורונים עלינו לבצע אימון (שזה החלק העיקרי של הפרויקט) ובדיקה של תהליך האימון.

כתוצאה מהחלוקה הנ"ל, גם הבדיקות יעשו על כל שינוי שיתבצע באחד מהחלקים.

למשל, אם נבצע שינוי באופן טעינה המידע או בנרמולו, נצטרך לבצע בדיקה שהביצועים של הרשת לא נפגעו. באותו אופן, אם נבצע שינוי במבנה רשת הנורונים, נצטרך לבצע בדיקת ביצועים של הרשת.

לרוב, הבדיקה המיידית תהיה לבדוק את אחוזי הדיוק (Accuracy) של הרשת ואת ערך ה-Loss שנובע מהרשת (אם ערך ה-Loss גבוה זה אומר שביצועי הרשת לא טובים ואחוז הדיוק נמוך).

יש לציין כי תוכנית הבדיקות בפרויקט זה היא אינה סטנדרטית, זאת כאמור כי המדדים הנבדקים הינם האינדיקציה להצלחה או לכישלון. חישוב המדדים הנ"ל מתבצע בכל מספר איטרציות בתהליך האימון ומעריכה את הצלחת הרשת בלמידת המידע. לכן אין צורך לבצע בדיקות יחידה ו/או קבלה עבור פרויקט זה.

גם בדיקת טיב האלגוריתם, דהיינו תהליך האימון, נקבע על פי המדדים Loss ו-Accuracy.

6. סקירת עבודות דומות בספרות והשוואה

האב טיפוס יתבסס על פרויקט דומה מהטכניון. בשונה מפרויקט זה שבוצע בעזרת מחלקה ב-Python שנקראת Lasagna, אנו נתבסס על המחלקה TensorFlow של גוגל המיועדת עבור פיתוח אלגוריתמי Deep Learning באופן יעיל ומבוזר יותר. בנוסף, נרחיב את כמות האיברים המסווגים מ-2 ל-4.

במאמר DeepOrgan (סעיף b2 בביבליוגרפיה), נבנתה רשת ConvNet עמוקה בעלת 5 שכבות קונבולוציה שמטרתה לזהות לבלב מתוך סריקת CT. שימוש ברשתות ConvNet זהה לפתרון שהצענו בפרויקט זה. עם זאת, במאמר נבנתה רשת בעלת 5 שכבות קונבולוציה, שהיא עמוקה בהרבה מהרשת שנבנתה עבור פרויקט זה. כמו כן, המדד שבו השתמשו בו במאמר הוא Dice Similarity Coefficient שמשווה שונות של שתי סגמנטציות (הסגמנטציה המקורית והסגמנטציה שנוצרה בעזרת רשת הנורונים). בפרויקט זה, הסתמכנו על הדיוק וההפסד (Accuracy, Loss) של הרשת בלבד.

במאמר 3D Deeply Supervised Network for Automatic Liver Segmentation from CT Volumes (סעיף b3 בביבליוגרפיה), נעשה שימוש דומה ברשת ConvNet עם 6 שכבות קונבולוציה כדי לזהות כבד. המאמר התמקד באופטימיזציה של הרשת ושיפור מהירות החיזוי (Prediction) ביחס לרשתות אחרות. הפתרון שהוצע במאמר זה הוא שימוש בשכבת Fully Connected Conditional Random Field אשר, על פי המאמר, משפר את זמן החיזוי משמעותית. בהשוואה לפרויקט הזה, השיטה זהה אך רשת הנורונים שבשימוש במאמר מורכבת יותר. כמו כן, במאמר נעשה שימוש ברשת נורונים שיוצרת לזהות כבד, כלומר לא נעשה פיתוח מעבר לזה, אלא רק אופטימיזציה של הקיים.

7. מסקנות מהמימוש ומהפרויקט

ראשית, מאחר ופרויקט זה הוא יותר מחקרי מאשר הנדסי, נלמדו דרכים וגישות שונות כדי לגשת לבעיה ולנתח אותה, בפרט בעית סיווג. ישנן מספר דרכים מעולם הלמידה החישובית (Machine Learning) כדי לפתור בעיות סיווג, כמו למשל רגרסיה לינארית או רגרסיה לוגיסטית. עבור תמונות, הדרך המקובלת לפתור את בעית הסיווג היא באמצעות רשתות נוירונים עם קונבולוציה (ConvNets).

בסופו של דבר, בפרויקט זה לא יצאה סגמנטציה כפלט סופי של המערכת. עם זאת, מומש פתרון אפשרי אשר ניתן להרחבה למספר גדול יותר של איברים. התשתית שנבנתה מתבססת על התיאוריה של תחום הלמידה העמוקה (Deep Learning), תת תחום של למידה חישובית, כך שנעשתה למידה מקדימה רבה טרם מימוש הפתרון.

המסקנות מהפרויקט, בתור פרויקט בהנדסת תוכנה, הם בעיקר תיאור של כל הסיכונים האפשריים שעלולים לפגוע בשלמות הפרויקט. כמו כן, חשוב לבדוק את התיעוד של מחלקות חיצוניות שבאות לידי שימוש בפרויקט, כלומר האם התיעוד מפורט וברור מספיק והאם המחלקה אכן מתאימה לצרכי הפרויקט. חשוב גם לבדוק אם ישנן מחלקות אשר מתאימות יותר עבור הפרויקט, ולא להסתמך על המחלקה מהאב טיפוס, שמבוסס על מחקר קודם.

בנוסף, כפרויקט מחקרי, יש לתעד כל שלב ושלב בתהליך הפיתוח והמחקר. במהלך הפרויקט, נשמרו מעת לעת קבצי לוג (log), גרפים ומשקולות. שמירת המשקולות אפשרה להמשיך את תהליך האימון מנקודה מתקדמת וליצור מעין checkpoint לתהליך האימון.

הפרויקט נעצר בנקודה שבה צריך להוציא סגמנטציה, על סמך הסיווגים שהתקבלו מרשת הנוירונים. מאחר והתבצעו הזזות על המידע, וכן הרשת מקבלת חלק מהמידע בכל פעם (batches) לצורך אימון, היה צורך לשחזר את הסדר המקורי של הפאצ'ים. תהליך זה לקח זמן רב יותר מהצפוי ולכן לא הגיע לסימומו. עם זאת, מי אשר ימשיך במחקר הנ"ל בנקודה זו, יוכל לנסות לבצע את השחזור, ולהוציא סגמנטציה כנדרש.

8. רשימת ספרות \ ביבליוגרפיה

a. ערכים והגדרות

Machine Learning: .i

https://en.wikipedia.org/wiki/Machine_learning

Artificial Neural Network: .ii

https://en.wikipedia.org/wiki/Artificial_neural_network

Convolutional Neural Network: .iii

https://en.wikipedia.org/wiki/Convolutional_neural_network

Artificial Intelligence vs Machine .iv

Learning Vs Deep Learning:

<https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai>

The Theory of Neural Network: .v

<https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Neuron/index.html>

b. מאמרים

VISCERAL: Evaluation-as-a-Service for .i

Medical Imaging:

https://www.researchgate.net/profile/Henning_Mueller2/publication/309310522_VISCERAL_Evaluation-as-a-Service_for_Medical_Imaging_VISCERAL_book/links/5808d82708ae63c48ff000f3.pdf

DeepOrgan: Multi-level Deep .ii

Convolutional Networks for Automated Pancreas Segmentation:

<https://arxiv.org/pdf/1506.06448.pdf>

3D Deeply Supervised Network for .iii

Automatic Liver Segmentation from CT Volumes

<https://arxiv.org/pdf/1607.00582>

Automatic liver tumor segmentation .iv

in follow-up CT studies using Convolutional Neural Networks:

<http://www.cs.huji.ac.il/labs/casmip/wp-content/uploads/2015/11/c133-2015-miccai15-workshop-liver-tumors.pdf>

c. תיעוד

TensorFlow API של תיעוד .i

https://www.tensorflow.org/versions/r0.12/api_docs/index.html

9. קישורים למערכת לניהול הפרויקט ובקרת תצורה

#	מערכת	מיקום
1	מאגר קוד	github.com/talshi/JCE-Final-Project
2	יומן	https://calendar.google.com/calendar/embed?src=ivrq6ko4m2tadb0oa9rplpc8h4%40group.calendar.google.com&ctz=Asia/Jerusalem
3	ניהול פרויקט (אם בשימוש)	אין
4	הפצה	AWS
5	סרטון אב-טיפוס	https://drive.google.com/file/d/0B_05KBce8SJ0TmljTXRjMXE1NTg/view?usp=sharing

10. תכנון הפרויקט

פגישה ראשונה עם המנחה האקדמי	26.5.2016
החלטה על נושא הפרויקט	26.7.2016
פגישה שנייה עם המנחה האקדמי והחלטה סופית על נושא הפרויקט מול המנחה	11.8.2016
תיאום ציפיות בין הסטודנטים	11.8.2016
פ"ע פרויקט גמר	26.9.2016
פ"ע פרויקט גמר	15.10.2016
פ"ע פרויקט גמר	4.11.2016
פ"ע פרויקט גמר	11.11.2016
פ"ע פרויקט גמר	13.11.2016
פ"ע פרויקט גמר	16.11.2016
פ"ע פרויקט גמר	18.11.2016
פגישה עם המנחה האקדמאי	20.11.2016
פ"ע פרויקט גמר	25.11.2016
פ"ע פרויקט - התנעת אב טיפוס	30.11.2016
פ"ע פרויקט - סגירת הצעה	2.12.2016
פ"ע פרויקט - פיתוח אב טיפוס	6.12.2017
פ"ע פרויקט - סגירת אב טיפוס	13.1.2017
פ"ע פרויקט גמר	27.1.2017
פ"ע פרויקט גמר	17.2.2017
פ"ע פרויקט גמר	3.3.2017
פגישה עם המנחה האקדמאי	7.3.2017
פ"ע פרויקט גמר	10.3.2017
פגישה עם המנחה האקדמאי	30.3.2017
פגישה עם המנחה האקדמאי	3.5.2017
פגישה עם המנחה האקדמאי	6.6.2017
פ"ע פרויקט גמר	9.6.2017
פ"ע פרויקט גמר	16.6.2017
פ"ע פרויקט גמר	18.6.2017

11. טבלת סיכונים

#	הסיכון	חומרה	מענה אפשרי
1	הערכה שגויה של גודל המערכת	נמוך	קריאה מקיפה של מסמכי האב טיפוס, הבנת המערכת על בוריה.
2	הערכה שגויה של לוח הזמנים	בינוני	קידום משימות ככל שניתן לפני מועד הסיום, תכנון מוקדם של לוח הזמנים תוך לקיחת מרווחי ביטחון ועמידה ביעדי הלו"ז.
3	קריאת תצלומי CT	בינוני	למידה מקדימה של הנושא, פגישה עם מומחים והתייעצות במידת הצורך.
4	היכרות עם טכניקות למידה חישובית ולמידה עמוקה	בינוני	למידה מקדימה ומעמיקה של הנושא. לקיחת קורסים בנושאים הנ"ל במידת הצורך.
5	שימוש ברשתות נירונים כפולה ומשולשת לא ישפר את אחוזי הדיוק ויבזבז זמן יקר בתהליך האימון	בינוני	למידה מקדימה ומעמיקה של הנושא. ניהול זמנים ותכנון מוקדם של חלק זה בפרויקט.
6	שימוש במחלקות שלא מתאימות למטרת הפרויקט	בינוני	חיפוש מחלקות אחרות אשר מתאימות יותר, שימוש ביותר ממחלקה אחת כאשר כל מחלקה מתאימה לחלק אחר של הפרויקט.

12. רשימת/טבלת דרישות (User Requirement Document)

מס' דרישה	תיאור
1	זיהוי מהיר שאורך עד 2 דקות.
2	רמת דיוק של 90% לפחות.
3	הרחבת יכולות המערכת לזיהוי של כ-4 איברים.

Abstract

Automatic organ classification is an important and challenging problem for
.medical image analysis

In recent years, techniques have been developed for various classification
problems in many fields, especially in medicine. Most techniques are based on
artificial neural networks, and in particular artificial neural networks with
.convolution layers, which are better in image analysis

In this project, we present a solution for classification of organs from CT scans
.using artificial neural networks (ConvNets)

The center of the project focuses on investigating the problem of classification
.by neural networks

We have a CT dataset, which our goal is to build a system that classifies CT scans
so that the system can analyze and obtain useful information from scans for
.classification purposes

We will detail the neural networks in general, detailing the stages of preparation
of the information and the stages in building the neuron network, detailing the
network we created, the training process and the results we received, as well as
.details of the difficulties that arose during the development stages

Software Engineering Department

Machine Learning for Medical Data - CT Images Analyzing

by

Guy Biton

Tal Shiran

:Academic Supervisor
Assaf Shpanier

Software Engineering Department

Machine Learning for Medical Data - CT Images Analyzing

by
Guy Biton
Tal Shiran

Sivan 5777

June 2017