**סיכום פגישות חן וברק**

עד התאריך ה-5.12:

כתבנו קוד ראשוני שמייצר לוח ומוחק ממנו מספרים כך שיהיה פתרון יחיד לסודוקו. הקובץ שומר שומר את הלוח המקורי ואת הלוח עם המספרים המחוקים בקובץ py.(זה מציג בג'יבריש כאשר עושים בקובץ py אולי נשנה ונעזר ב-numpy) נעזרנו באתר:

<https://www.101computing.net/sudoku-generator-algorithm/>

קראנו לקובץ: createGrid

הצלחנו לשנות את לולאת ה-while כך שיקח מהר יותר (שלא יחפש באופן אקראי מספר מהלוח).

ניסינו לעשות את הקוד כללי ללוח לא בהכרח 9 על 9 אבל עדין לא הצלחנו.

כדאי אולי לשנות את הקוד שיהיה יותר יפה וגם שיעבוד על לוחות שהם לא 9 על 9.

בנוסף, כתבנו קוד שפותר את הלוח באמצעות backtracking . נעזרנו באתר:

<https://www.geeksforgeeks.org/sudoku-backtracking-7/>

קראנו לקובץ: solveBacktracking

כתבנו דוח אפיון שבו התייחסנו לסקר הספרות שאותו נבצע.

<https://www.tandfonline.com/doi/full/10.1080/10586458.2013.870056>

McGuire, Gary, Bastian Tugemann, and Gilles Civario. "There is no 16-clue Sudoku: solving the Sudoku minimum number of clues problem via hitting set enumeration." Experimental Mathematics 23.2 (2014): 190-217.

מאמר שמסיר שיש מינימום 17 מספרים בסודוקו ואומר כמה אפשרויות יש ללוחות סודוקו.

התחלנו לכתוב סקר ספרות ודוח אפיון:

אתרים באינטרנט שמצאנו:

<https://www.tandfonline.com/doi/full/10.1080/10586458.2013.870056>

McGuire, Gary, Bastian Tugemann, and Gilles Civario. "There is no 16-clue Sudoku: solving the Sudoku minimum number of clues problem via hitting set enumeration." Experimental Mathematics 23.2 (2014): 190-217.‏

<http://micsymposium.org/mics2018/proceedings/MICS_2018_paper_17.pdf>

<https://www.mathworks.com/company/newsletters/articles/solving-sudoku-with-matlab.html>

<https://www.mathworks.com/help/optim/examples/solve-sudoku-puzzles-via-integer-programming.html>

<https://www.kaggle.com/bryanpark/sudoku>

<https://ieeexplore.ieee.org/abstract/document/7333884>

<https://ieeexplore.ieee.org/abstract/document/7007986>

קוד מחולק לפי רמות שהגיע ל-86% דיוק (כאן יש חלוקה לרמות קושי):

<https://github.com/Kyubyong/sudoku>

פתרון שהגיע ל-99.7% ללא קונובלוציה רק fully connected:

<https://www.kaggle.com/dithyrambe/neural-nets-as-sudoku-solvers>

פתרון שהגיע ל-99 אחוז:

<https://towardsdatascience.com/solving-sudoku-with-convolution-neural-network-keras-655ba4be3b11>

<https://github.com/shivaverma/Sudoku-Solver-CNN/blob/master/sudoku.ipynb?source=post_page-----655ba4be3b11---------------------->

10.12.19

נפגשנו עם רון ותום וסיפרנו להם את הפתרונות שמצאנו.

דברים מרכזיים שעלו מהפגישה:

* להתחיל לכתוב קוד.
* לבדוק אם ה-loss הוא עבור המשבצת החסרה לבד בכל פעם או יותר.
* לבדוק את התוצאות של הפתרון שראינו אולי על לוחות קשים יותר (יותר מ-50 משבצות חסרות)
* לנסות לעשות קונבולוציה במקום הפתרון שהגיע ל-99.7% עם fully connected בלבד.
* לעשות One hot עם 9X9X9 או 9X9X10 (ספרה נוספת עבור האפס)
* להתחיל מהפתרון שראינו באינטרנט
* לא מנרמלים one hot
* לא בהכרח חייבים להשתמש במליון לוחות מהדאטה. להשתמש במספר לוחות שיביא לנו תוצאה טובה.
* אפשר להשתמש ב-keras או ב-pytorch או ב-tensorflow (מה שבא לנו).
* כשנקבל רשת שלמדה - דרך כלשהי להבין מה הרשת למדה ולנתח את התוצאות:

אפשר לבדוק איזה מספר הוא בעל הסתברות מקסימלית לצאת. כלומר אם ההסתברות הכוללת של 4 למשל יותר גבוהה מההסתברות הכוללת של 9 עבור קלט מסוים (עם מעט 4 למשל).

12.12.19

סיימנו לכתוב דוח אפיון ושלחנו אותו. הכנסנו 2 אתרים מהאתרים שמצאנו והכנסנו מאמר אחד לסקר ספרות.

25.3.20

בקובץ Sudoku\_CNN\_1 יש לנו רשת FC שעובדת כבר. מחקנו במקסימום 20 מספרים בקפיצות מסוימות:

 for n\_epochs, n\_delete in zip([1, 1, 2, 1, 2, 1, 2, 3, 3, 2, 2, 2, 2],

                                  [0, 1, 2, 3, 4, 6, 8, 10, 12, 14, 16, 18, 20]):

יש לנו רשת CNN שעובדת כבר . מחקהו במקסימום 25 מספרים בקפיצות מסוימות:

for n\_epochs, n\_delete in zip([1, 2, 2, 1, 1, 1, 3, 3, 3, 5, 5, 10, 10],

                                  [0, 1, 2, 3, 4, 6, 8, 10, 12, 13, 15, 20, 25]):

משימות שחשבנו עליהן:

1. לעשות גרפים של התוצאות ו-confusion\_matrix.
2. לנסות לשחק עם ה-loss שמותאם לבעיה שלנו (עמודה,שורה,ריבוע).
3. למחוק עוד מספרים ולהגיע לתוצאות טובות. (אולי למחוק בקפיצות יותר קטנות)
4. ב-test לעשות בדיקה לפי כמה משבצות ריקות יש בלוח.

29.3.20

יש לנו 2 קבצים כרגע שעובדים:

Sudoku\_FC – רשת Fully connected שבכל איטרציה מוחקת מספר (או כמה מספרים) ואז מתאמנת על הלוחות המחוקים. כרגע הרשת פותרת את כל הלוח בבת אחד ולא משבצת משבצת. אחרי בערך 30 מספרים שמחקנו הגענו לבערך 93 אחוז דיוק.

Sudoku\_CNN – אותו הדבר כמו רשת ה-FC אבל עם במקום FC, רק שהשתמשנו בשכבות CNN במקום. גם פתרון זה פותר בבת אחת את כל הלוח. המסנן בגודל של שורה, עמודה וריבוע.

עכשיו ננסה לפתור את הלוח בצורה כזאת שהרשת תפתור משבצת-משבצת בכל פעם (גם בשלב באימון וגם בשלב ה-test)

הפתרון שאיתו אנחנו עובדים:

<https://www.kaggle.com/dithyrambe/neural-nets-as-sudoku-solvers>

30.3.20

היה לנו בעיה בפונקציית הטרנספוז: כאשר השתמשנו בטרנספוז של np הייתה לנו בעיה של המרה מ-np ל-torch. כי אז האובייקטים לא היו בעלי תכונות גראדינט מתאימות. לכן החלטנו להשאר ב-torch ולעשות טרנספוז באמצעות torch.

2.4.20

פגישה עם המנחים:

Masking על הloss .

לא לעשות epoch עבור 0 מספרים שמורידים.

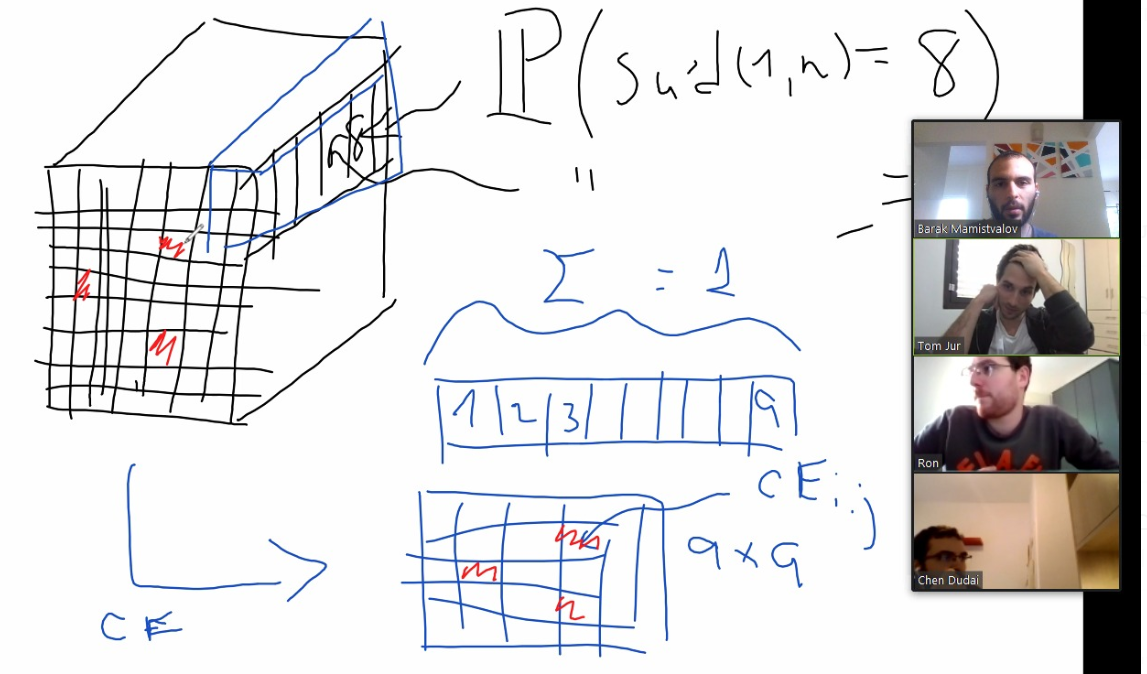
Cross entropy לכל תא בכניסה – ולא binary cross entropy. רוצה לעשות את זה על axis מסוים. ה-loss שיחזור הוא הסכומים של ה-cross entropy (( 9X9. ואז עושים mask. Loss size יהיה סקלר.

לעשות upload לקוד שכתבתנו ב-pycharm וככה יהיה יותר נוח לעבוד ב-colab.

לעשות מספר epochs אינסופי כממשיכים כל עוד מתקדמים. לנסות משהו פשוט בהתחלה.

פעולה שנקראת detach שמנתקת את הטנזור מהגרף חישוב שלנו.

הבנו שכן אנחנו צריכים למלא ב-train את הכל בבת אחת ולא אחד-אחד. רק בtest נמלא אחד-אחד. נשתמש ב-mask בtrain.



פרמול של ברק:

עד כה עבדנו עם binary\_cross\_entropy לוס שלמעשה מבצע לוס איבר איבר במתוך המטריצה התלת מימדית.

הבנו מתוך ךהשיחה עם רון ותום, שלמעשה יותר נכון יהיה לבצע cross\_entropy לוס על כל וקטור וואן הוט בנפרד, כך למעשה נקבל מטריצה דו מימדית של איברי הלוס המתאימים לכל וקטור כזה.

בשביל שנוכל לקנוס את הרשת רק על איברים חסרים, השתמשנו במכפלה של המטריצה הזו במסכה של המספרים הסחרים ופעפענו את הלוס אחורה. במצב כזה בכל מקום בו האיבר לא היה חסר, הלוס יהיה 0 ולכן המשקולות לא יושפעו מתוצאות במקומות של איברים שלא היו חסרים.

21.4

לוקח לרשת המון זמן ללמוד. החל מ-14 מספרים מחוקים התוצאות להיות פחות טובות מ-100 אחוז.

באזור ה-60 מספרים מחוקים מגיעים ל-0.87 דיוק (כשיש גודל צעד 0.001 בלי שינוי, מקסימום epochs 10, קפיצות של 2).

יש כמה הצעות שחשבנו עליהן על מנת לשפר את התוצאות של הרשת:

1. לעשות רשת קונובלוציה (במקום FC שעשינו עד כה)
2. לשנות עוד את גודל הצעד
3. לאמן ככה שה-train יבוצע כמו הולדיציה ו- test.
4. Ensemble learning

בדיקה שעשינו:

60 מספרים מחוקים (בלי איטרציות בבת אחת) ב-FC הגענו ל- 0.839 דיוק על סט הולדיציה אחרי 86 איפוקים.

3.5.20:

פגישה עם המנחים –

לעשות גרף שמראה איך אחוז הדיוק יורד ככל שמוחקים יותר מספרים.

זה בסדר שהאימון והולדיציה לא מתבצעים באותו אופן.

לא לעגל את התוצאה של הולידציה – ככה אפשר לראות בכמה לוחות נכשלנו ולמה.

להעריך כמה זמן מבלים על ה-test - אולי לא לעשות ולידציה כל epoch, או לעשות ולידציה על סט יותר קטן.

שורה 61 – לחלק ב- sum of deleted cells

שורה 78 – להסתכל על ההדפסה.

לא לקבוע את מספר ה-epchos מראש, אלא להשתמש ב-patience.

לעשות checkpoint ו- restore.

לשמור ביחד עם המודל שאנחנו שומרים את תנאי הריצה.

אפשר לעבוד מרחוק על המעבדה – לדבר עם קובי

שומרים את ה-best model – אם אין התקדמות בולידציה, לטעון את הטוב ששמרנו ואז להמשיך.

לקבע את הלוח – והראות כמה הפרדיקציה משתפרת:

להשוות מול ה-baseline - כמה אנחנו פחות טובים ממנו.

אם רוצים אקסטרה נקודות:

לעשות embedding של הפיצ'רים האחרונים על ידי TSNE.

לעשות TSNE (או על ויאליזציה אחרת) על וקטור באורך 729 – ולהשוות בין מה שצדקנו, למה

שטעינו.

לחשוב על כל מיני ויזואליזציות מעניינות שיעזרו לנו להפיק תובנות.

לתת שם אחר ל-ACC של ה-train.

לרשום ACC של הולידציה בבת אחת, בנוסף ל-ACC של מחיקה מספר מספר.

ב-CNN ה-learning rate אולי אחר, צריך גם פה לעשות מספר epochs לא מוגבל.

דברים שצריך לעשות:

לעבוד על זה שהאימון יקח פחות זמן – לדבר עם אורלי, פחות ולדיציות.

להבין מה נקודת העצירה שאנחנו רוצים – אולי אפשר כבר לעצור.

לחשוב על ויזואליזציה לאורך הדרך.

לבדוק CNN לעומת FC. הציפייה היא ש- CNN יעבוד טוב יותר מ-FC.

ברק העלה את ב-ensemble learning – לשים לב שעושים את זה על אותה רשת! לא על רשתות שונות. למשל מאמנים כמה פעמים רשתות FC (או כמה פעמים רשתות CNN) ואז משתמשים באותן רשתות. לעשות את ה-ensemble רק אם זה משפר את התוצאות בלי יותר מדי השקעה.

לשים לב שצריך להתחיל להתכנס למה שעובד, אם יש משהו שיכול לשפר ולוקח יותר מדי זמן אולי לא שווה להשקיע בו.

8.6.20

ראינו שיש בממוצע 47.2 איברים מחוקים ב-baseline.

ב-baseline מגיעים ך-99.7% לוחות שפותרים לגמרי. הטסט שלנו קצת שונה לשים לב. הוא בודק כמה לוחות נפתרו בצורה מושלמת.

ההיסטוגרמה של הלוחות המחוקים ב-baseline – סה"כ 10,000 לוחות:

108 לוחות עם 45 מספרים מחוקים

1959 לוחות עם 46 מספרים מחוקים

4499 לוחות עם 47 מספרים מחוקים

2739 לוחות עם 48 מספרים מחוקים

603 לוחות עם 49 מספרים מחוקים

87 לוחות עם 50 מספרים מחוקים

5 לוחות עם 51 מספרים מחוקים

אנחנו הגענו ל99.2% לעומת 99.7% של ה-baseline.

אולי להציג היסטוגרמה בדוח של המספרים הנ"ל

בכל הרשת שלנו עשינו leaky relu.

מעניין יהיה לראות איך הרשת המאומנת תהיי עם ReLU במקום (או עם פונקציית אקטיבציה אחרת).

לעשות גרפים של התוצאות(train,test,Val) ו-confusion\_matrix: עבור FC,CNN,Ensamble. לבדוק את התוצאות עבור מספר שונה של רשתות Ensable.

גרפים של התוצאות של ה-BaseLine ותוצאות של הטסט של הBaseLine עבור FC,CNN,Ensamble.

היסטוגרמות של מספר הטעויות בכל לוח.

אולי להוסיף dropout כדי לפר את התוצאות – כי ה-baseline עשו dropout.