יום שלישי 28 פברואר 2023

למידה עמוקה – חורף – 203.3834

פרויקט בנושא רשת נוירונים סיאמית שקובעת אם שני ציורים שונים מגיעים מאותו אמן

מרצה: ריטה אוסדצ'י

אלון פפיני

205815962

ליאור מלץ

318307923

מטרה:

מטרת הפרויקט שבחרנו היא לבנות רשת קונבולוציה שיודעת לקבל כקלט שני קבצי תמונה המייצגים ציור של אמן כלשהו, ולקבוע (באחוזי דיוק סבירים, לפחות) בסוף התהליך האם שני הציורים הנ"ל צוירו על ידי אותו אמן, או על ידי אמנים שונים.

בפרט, נושא הפרויקט הוא למעשה התחרות בעל השם Painter by Numbers באתר Kaggle, שהסתיימה בשנת 2016. להלן הלינק לתחרות:

https://www.kaggle.com/competitions/painter-by-numbers

נתונים:

הנתונים שבעזרתם מימשנו את הרשת הם הקבצים שזמינים בעמוד ה-Kaggle הנ"ל. ספציפית, נעזרנו במגוון קבצי ה-train (גם בקובץ הגדול וגם בקבצים הממוספרים, שמהווים חלוקה של קובץ האימון הגדול) ו-test בשביל עשרות אלפי תמונות ה-test וה-training, ובקובץ all_data_info.csv על מנת לחלץ מידע רלוונטי על כל תמונה – מספר מזהה, האמן שצייר אותה, ממדי התמונה וכו'. מעבר לכך, כחלק מעבודת ה-pre-processing שלנו כתבנו פונקציה שתעשה scale לכל התמונות שאנחנו עבדנו איתן לרזולוציה קבועה וקטנה יותר מהממוצע של ממדי התמונות על מנת להקל ולקצר את תהליך האימון של הרשת שלנו. במהלך מרבית העבודה על הפרויקט קבענו שהרזולוציה שאיתה נעבוד היא 256x256, וכל תמונה "נורמלה" ללהיות בעלת 3 ערוצים: RGB (כלומר, תמונות מסוימות שהיה להן ערוץ רביעי של alpha איבדו את הערוץ ההוא, ותמונות greyscale הורחבו למרחב ה-RGB).

ארכיטקטורת הרשת:

פה מתחיל החלק שבו התנסינו במגוון רב של פרמוטציות. הבסיס לרשתות שניסינו הגיע מהרשת שניתן למצוא בלינק הבא, שבעזרתו גם מימשנו את פונקציה ה-triplet loss שלנו:

https://www.kaggle.com/code/hirotaka0122/triplet-loss-with-pytorch/notebook

אחרי איטרציות רבות ומגוונות על הרשת הזאת, הגענו לגרסה שהניבה תוצאות לא רעות בשבילנו: הרשת, שהיא בעצם רשת CNN עם סוף שהוא fully connected התחילה עם חלק סיקוונציאלי קונבולוציוני שכלל ארבע שכבות קונבולוציה הכוללות פונקציית ReLU, פעולת MaxPool וסיכוי גם ל-Dropout כדי להימנע מ-overfitting. לאחר החלק הזה הרשת המשיכה בחלק fully connected, כאשר אחרי חלק fc אחד היה ReLU ואחרי חלק fc שני ואחרון היה עוד ReLU אחד. נדגיש שהתיאור הנ"ל הוא רק של הרשת שהגענו אליה לפני ההגשה, אך למעשה במהלך העבודה על הפרויקט המבנה עצמו של הרשת השתנה – הוספנו שכבות, הסרנו שכבות, ובאופן כללי גיוונו בתצורה שלה. בפרט, ביצענו שינויים מרובים ברשת שלנו שכללו את שינוי ההיפר פרמטרים של גדלי המימדים של הגרעינים השונים, סיכויי dropout וגם הוספנו (ו/או הסרנו במקרים אחרים) נורמליזציה של ה-batch-ים ששלחנו לאימון במודל שלנו. במהלך האימונים הראשוניים שלנו שמנו לב כיצד הגדלת הגרעינים הקונבולוציוניים, דבר שהקטין את ממדי התוצאה, עזר במהלך אימון הרשת להגיע לערכי loss נמוכים יותר מהר יותר. אמנם זה לא בהכרח אומר שהגענו לתוצאות דיוק טובות יותר...

בסוף הפרויקט המודל שלנו שינה צורה ל-"סופר מודל", שבעצמו הכיל 10 תתי-מודלים מאומנים מראש (כאשר הרשת שלהם בנויה כפי שתיארנו לעיל), וכל אחד מתתי-המודלים בעצמו אומן על כ-20 מהאומנים (והציורים השייכים להם) שלהם הכי הרבה ציורים. בסך הכל מודל העל מורכב מתתי-מודלים שאומנו על כמעט 200 אמנים שונים. זאת הייתה הדרך שלנו להתגבר על מגבלות הרשת המקורית באימון על מספר כל כך רב של אומנים שונים, ומספר גדול אף יותר של ציורים שונים.

שיטת בדיקה ואימון המודל:

ועכשיו, לחלק שלקח את רוב הזמן – אימון המודל. לשם אימון הרשת, מימשנו בעצמנו dataloader במספר גרסאות, כאשר כל גרסה נועדה לתמוך בצורה ספציפית של חישוב פונקציית loss. מכיוון שמדובר ברשת סיאמית, שתי פונקציות ה-loss הרלוונטיות הן פונקציית contrastive loss ופונקציית triplet loss. אחרי התנסות קצרה כבר בשלבים הראשוניים מצאנו את עצמנו נמשכים לפונקציית ה-triplet loss בשביל מאמצי האימון של הרשת שלנו. תזכורת: ב-triplet loss יש 3 תמונות כל פעם – תמונת העוגן, התמונה ה-"חיובית" והתמונה ה-"שלילית". השאיפה היא להגיע למצב שבו תמונת העוגן היא קרובה ביותר לתמונה החיובית (שצוירה על ידי אותו אמן כמו תמונת העוגן) ורחוקה ביותר מהתמונה השלילית (שצוירה על ידי אמן אחר). הרשתות שהתנסינו איתן ידעו לקחת תמונה בגודל 3x256x256 ולהוציא עבורה feature vector, שבתורו שלחנו (בשלב האימונים) לבדיקה בפונקציית ה-loss שבחרנו כדי לראות כמה קרובים הווקטורים של התמונה לווקטורים של התמונה החיובית, וכמה רחוקים הם מהתמונה השלילית (לפי הנוסחה המקובלת בפונקציית triplet loss), ואימון הרשת כלל gradient steps שנועדו לשפר את תוצר הרשת ולהשאיף את ערך ה-loss ל-0.

על מנת לשפר את תהליך האימון הוספנו אופטימיזציות כמו שימוש ב-Adam, Learning Rate Scheduler שידעו לשנות את קצב הלמידה לפי הטרנדים של תוצאות דיוק הרשת אחרי כל epoch. התנסינו גם עם אימון הרשת על חלקים נפרדים של תמונות ה-test (תוך כדי ווידוא שתמונות ה-validation הן שונות לגמרי מאלו של ה-test), ואף הוספנו תנאי patience כדי לדעת לעצור את האימון אחרי שלב מסוים של חוסר שיפור מוחלט.

ניסיון נוסף בשיפור תהליך האימון היה משהו שקראנו לו hard batches. חוץ מהעובדה שניסינו למקבל ולשפר את האימון על ידי שימוש במספר inputs בכל batch, ניסינו גם לבנות batches "קשים" – לקחנו תמונות מתוך batch וניסינו ליצור מהן triplets בינן ובין עצמן באופן כזה שהמרחק בין העוגן והתמונה החיובית הוא הגדול ביותר שאפשר, והמרחק שבין העוגן והשלילית הוא הקטן שאפשר. באופן זה קיווינו שנוכל "להכריח" את הרשת להשתפר בקצב מהיר יותר, על ידי כך שכפינו עליה קלטים קשים, שבהכרח ייתנו גראדינטים גדולים.

במהלך ניסיונות האימונים השונים שלנו ניסינו לשנות את מספר ה-batches שהעברנו במודל בכל epoch (כאשר בכל epoch אתחלנו מחדש את ה-dataloader, בניסיון להימנע מ-overfitting), ובשלבים מאוחרים יותר הוספנו גם plots של דיוקים ואחוזי loss של training ושל validation במהלך ה-epochs של האימון שהודפסו בסוף תהליך אימון. כדי להוסיף אלמנט של ויזואליזציה תוך כדי האימון הוספנו גם פונקציה שנעזרה ב-PCA כדי לייצג feature vector של תמונה כנקודה תלת-ממדית, וכך הדפסנו בגרף נקודות שונות שנצבעו בצבעים המייצגים אמנים שונים כדי לראות כיצד הם ממוקמים אחד ביחס לשני. בשלב מוקדם של הפרויקט, כאשר אימנו את המודל על מספר מצומצם של אמנים, ראינו במקרה אחד של overfitting כיצד מרבית הנקודות עם אותו הצבע ממש הצטברו ב-cluster שהיה כמעט ו-"לא מזוהם" בנקודות מצבע אחר, שזה בערך מה שחיפשנו בהתחלה.

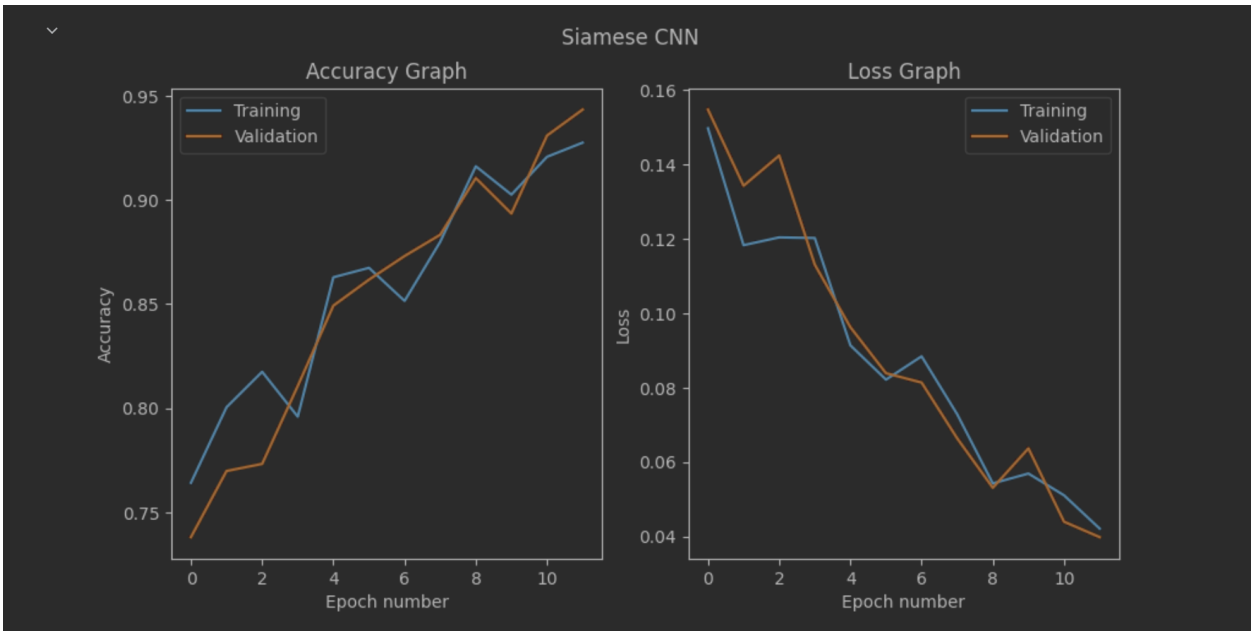
לקראת הסוף, כאשר החלטנו שברצוננו להפוך את המודל למודל על של 10 תתי-מודלים, הפרדנו את ה-200 (+-) אומנים בעלי מספרי הציורים הכי גדולים, ואימנו בנפרד 10 רשתות שונות על הציורים שלהם,

כפי שתיארנו בסוף הסעיף של ארכיטקטורת המודל. כל אחת מתתי-המודלים בנפרד הגיע לאחוזי דיוק טובים מאוד בשלב ה-training, אז סברנו שגם אם הם הגיעו למצב של overfitting לאומנים שעליהם הם אומנו, מודל-העל יוכל לנצל אותם בצורה מועילה: הוא שולח את ה-input שלו לכל אחד מתתי-המודלים ויוצר feature vector בסוף שהוא concatenation של כל הפלטים של תתי-המודלים. מכאן, קיווינו שאחוזי הדיוק בתהליך אימון הסופר-מודל יצדיקו את הפיצול הלא קונבנציונאלי הזה.

תוצאות:

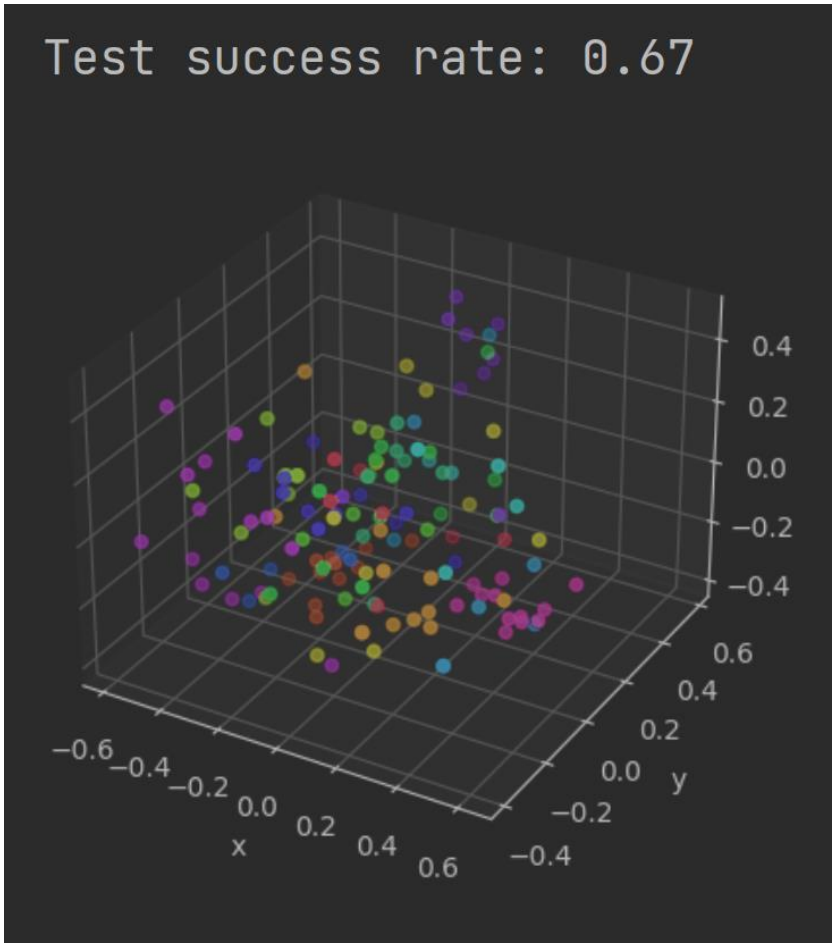
אחרי ניסיונות מרובים באימונים של כל מיני גרסאות בכל מיני גדלים של רשתות שהרכיבו את המודלים שלנו, להלן כמה תוצאות שהגענו אליהן:

הנה אימון של אחת מתתי-המודל שמרכיבים את סופר-המודל, שהגיע לתוצאות לא רעות מבחינת דיוק על מבחר מוגבל של אמנים -



לאחר מכן, הנה תוצאת האימון של מודל-העל, שמורכב מ-10 תתי-מודלים שאומנו מראש:

לבסוף, הגענו לתוצאה שהיוותה תקרת הזכוכית שלנו במקרי הקיצון של העולם האמיתי – כאשר ביצענו אבליואציה של המודל הסופי שלנו על ה-test dataset. נדגיש שממה ששמנו לב, אין הרבה ציורים ו/או אמנים שייחודיים ל-dataset של ה-test, כך שלהגיע לאיזשהו אחוז גבוה של דיוק איתו הוא אתגר אף יותר רציני מלהגיע לאחוז דומה ב, לדוגמה, evaluation:



הוראות הפעלה והסברים נוספים:

בשל מגבלת זמן, אנחנו נאלצנו להעביר את הוראות ההפעלה ואת ההסברים הכלליים הנוספים על ההגשה שלנו לסרטון, שאותו נשלח באיחור קליל לעומת הגשת הפרויקט. אנחנו מקווים שאפשר לסלוח על כך, ונשתדל שהסרטון יהיה בר צפייה 😊

נ.ב. להלן לינק לגיט שיצרנו בשביל הפרויקט. אנא שימו לב שבגלל ה-GITIGNORE אין בו את הקבצים המאסיביים של המודלים המאומנים או של הדאטאסטים שבקצבי זיפ, אבל היסטוריית ה-commits שלנו ביחד עם המבנה הכללי של קבצי הפרויקט נמצאים שם:

https://github.com/LiorMaltz/DeepLearningProject

תודה רבה על הקורס, ומקווים שהכוונות הטובות מאחורי הפרויקט יזרחו יותר חזק מחסרונותיו!