ראיה, תב 2.

<u>שאלה 1</u>

1. דגימה של 5 תמונות אקראיות מסט האימון:

ניתן לראות שהספרות לא פשוטות לזיהוי, כנראה מהסיבה שנלקחו מ"סצנות אמיתיות" (google maps).

2. תיאור הרשת שלנו לטובת פתרון בעיית הסיווג :

- הרשת שלנו מורכבת מ – 10 שכבות: 8 שכבות קונבולוציה, ו-2 שכבות FC .
-

- כמו שלמדנו ומניסיון קודם, השתמשנו בפונקציית אקטיביציית ReLu בגלל היתרונות המוכרים (גרדיאנטים לא נעלמים, חישוב מהיר יחסית וכו'):

$$f(x) = x^{+} = \max(0, x)$$

- בנוסף, פונקציית המחיר (הפונקציה הלא לינארית של השכבה האחרונה) היא Cross Entropy , מכיוון שמדובר בבעיית סיווג ל10 מחלקות:

$$\text{loss}(x, class) = -\log\left(\frac{\exp(x[class])}{\sum_j \exp(x[j])}\right) = -x[class] + \log\left(\sum_j \exp(x[j])\right)$$

- גדלי הפילטרים (גרעין הקונבולוציה) שהשתמשנו בהם הם 3X3XC לכל שכבות הקונבולוציה (כאשר C זהו גודל הערוצים שנוצרו בשכבה הקודמת, ובשכבה הראשונה זה 3 משלושת ערוצי הצבע). הסיבה לכך היא שתמונות הקלט שלנו הן בגודל 32X32X3 (יחסית קטן), ולכן גודל של 3X3 יצור receptive field יחסית משמעותי מהתמונה המקורית במקרה הזה לדעתינו.
- השתמשנו ב-2 שכבות FC ב-2 השכבות האחרונות של הרשת (כמתואר בהתחלה), ע"מ להוריד את המימד למימד הרצוי במוצא (במקרה שלנו 10 כמספר מחלקות הסיווג).
גדלי שכבות ה FC:
  - בשכבה הראשונה גודל הכניסה הוא 2048 (גודל מוצא השכבה שלפניה), ואילו גודל המוצא הינו 512 (החלטה הנדסית שלנו).
  - בשכבה השנייה גודל הכניסה הוא 512 (גודל המוצא של שכבה לפני), ואילו גודל המוצא הינו 10 , כמספר המחלקות לסיווג (המוצא הינו softmax – וקטור באורך 10 של הסתברויות למחלקות).
- גודל הקלט לרשת הוא גודל התמונה ב dataset : 32X32X3 .
גודל המוצא של הרשת הוא וקטור הסתברויות לתיוג של כל מחלקה : 10X1 .
- כמו שראינו בתרגול, מספר המשקולות (פרמטרים נלמדים) ברשת שלנו הוא:

```
#number of total trainable params of the model:
total_params = sum(p.numel() for p in model.parameters() if p.requires_grad)
print("Total trainable params : ",total_params)
```

```
Total trainable params :  3400138
```

3. ע"מ לאמן את המודל שלנו פיצלנו את ה test set המגיע מובנה עם SVHN dataset כך ש60% ממנו ילך ל test set שלנו, ואילו 40% ילך ל validation set . סה"כ נקבל שסך המידע שלנו יתחלק בערך כך: 73% אימון , 10% ואלידציה, 17% בדיקה.
את כוונון ה-היפר פרמטרים שלנו ביצענו בעזרת סט הואלידציה עד שהגענו לערכים מספקים.
נתוני האימון :

- גודל ה batch – 64.
- Learning rate = 1e-4.
- #Epochs = 20.
- Optimizer = Adam.
- בחרנו להוסיף אוגמנטציות אפיניות רנדומליות לסט האימון

```
transforms.RandomAffine(degrees = 25, translate=None, scale=(0.75,1.5),
                        shear=5, resample=False, fillcolor=0),
```

הסבר לאוגמנטציות:

- Degrees=25 – כל תמונה מסט האימון תקבל זווית רנדומלית לסיבוב בין 25- ל 25 מעלות (זווית גדולה מדי הייתה לא הגיונית ביחס לכך שהסט מדבר על ספרות).
- Scale=(0.75,1.5) – כל תמונה מסט האימון מקבל ערך רנדומלי בין 0.75 ל 1.5, כך שהתמונה מקבלת scale עם הגודל הרנדומלי שמתקבל.
- Shear=5 , כל תמונה נחתכת על ציר ה-X בערך רנדומלי של עד 5 פיקסלים מימין ומשמאל.

*כל האוגמנטציות הנ"ל הגיוניות לסט תמונות שמייצג ספרות, ולכן מוסיפות רנדומליות ורובסטיות לרשת שלנו.

- אחוז הדיוק עבור סט האימון (שלנו) –

```
test accuracy: 95.583%
```

אחוז הדיוק עבור סט הואלידציה (שלנו) –

```
valid accuracy: 95.409%
```

*ולכן אם רוצים את תוצאת הדיוק עבור סט האימון המקורי המגיע עם ה dataset נקבל:

$$\boxed{0.6 \cdot test\_accuracy + 0.4 \cdot validation\_accuracy = 95.51\%}$$

*במחברת הקוד יש אחוז דיוק קצת שונה, מכיוון שלאחר שסיימנו את העבודה הרצנו מחדש על CPU ואז שוב על GPU ע"מ לראות שהכל רץ חלק. ההבדל הוא מינורי..

שאלה 2

1. טענו בקוד.
2. התמונות:

3. הצעדים שביצענו ב pre-process לפני הכנסה למודל:

- ביצענו resize ל 224X224 כמו שכתוב באתר של models ב pytorch, ע"מ ליצור תמונה שמתאימה לכניסה למודל (מעל 224X224).
- התאמה של העמודות כך שמימד הערוצים יהיה תואם ל torch.
- Scale של הערכים ל [0,1].
- נירמול לפי הערכים הנתונים בשאלה (ונתונים ב model zoo), כך שיתאים למודל.
- התאמה של המידע בכניסה למימדי הרשת (הוספת מימד ה batch).

4. *אנו מניחים כי אתם מתכוונים להציג את התוצאות הסופיות, ולא את וקטור התוצאות (שהוא ארוך וקשה להבין ממנו מה הסיווג הסופי).

בהתאם לתיוגים שמפורטים כאן :

https://gist.github.com/yrevar/942d3a0ac09ec9e5eb3a

העתקנו את הקוד אלינו כדי שנוכל להציג את ה Label, התקבל :

94 - hummingbird

90 - lorikeet

5. הכנסנו תמונה של הכלב שלי ("טיטו"), ביצענו pre process כמתואר ב2.3 וזו התמונה עם התוצאה שהתקבלה :

168 - redbone

מגוגל, redbone זה הכלב :

ולכן נראה כי התוצאה הגיונית.

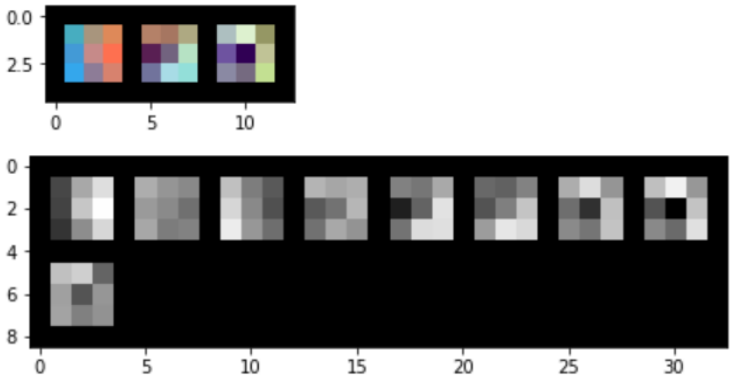
6. בחרנו פליפ אופקי , תמונת BGR (החלפה בערוצים) ופילטר גאוסיאני עם גרעין בגודל 5. התוצאות:

7. לאחר pre processing מתאים (2.3), נקבל את הסיווג :

ניתן לראות שעבור הפליפ והפילטר הגאוסיאני הסיווג נשמר, ואילו עבור שינוי מרחב הצבע הסיווג השתנה לכלב מסוג Weimaraner - מצ"ב דוגמא של כלב מסוג זה:

ביחס לתמונה המתקבלת במעבר ל BGR , הסיווג הזה הגיוני שכן "טיטו" דומה לכלב הזה רק בצבעים שונים.

8. נעזרנו בקוד מהתרגול, והצגנו את הפילטרים ב RGB (שמרנו על שלושת ערוצי הפילטרים) וב gray scale (ע"י הפרדת הערוצים) :



תגובת שלושת הפילטרים לארבעת התמונות:

מה שאנחנו רואים אלו התוצאות שמתקבלות במעבר של התמונות בכל אחד מ-3 הפילטרים הראשונים בשכבה הראשונה. ניתן לראות שהפילטרים מדגישים בעיקר קווי מתאר כללים של הדמות (שפות), מה שמתכתב עם העובדה שהשכבה הראשונה ברשת לומדת בעיקר פרטים כלליים כגון שפות.

*נעזרנו בקוד של ויזאוליזציה של שכבה עם שינויים שלנו, ורק לאחר מכן ראינו שיש פונקציה לויזואליזציה של פילטרים. מכיוון שזה עובד ונכון השארנו כך (עם הפונקציה שלנו).

9. בחרנו את שכבת ה FC האחרונה, עם מימד מאפיינים של 1000 (לכל תמונת כניסה יש 1000 מאפיינים לאחר המעבר בכל השכבות ב VGG16, כולל השכבה האחרונה).

המשך ←

10. השתמשנו ב linear SVM כמו שהוצע.

ניתן לראות כי המסווג הצליח לתייג את כל התמונות.

התוצאה הגיונית מכיוון שאנו יודעים שה"חכמה" ברשתות נוירונים היא הלמידה של המאפיינים (במקום feature engineering שהיה נהוג בשיטות הקודמות), וכי הפעולה האחרונה ברשתות נוירונים היא פעולת סיווג פשוטה. מכאן, שאם חילצנו את המאפיינים לאחר הלימוד של הרשת והכנסנו לSVM זה דיי שקול לשימוש ברשת לכל פעולת הסיווג.