# למידה חישובית 2

תרגיל בית מס': 2

שם: יונתן קויפמן

ת"ז: 212984801

תאריך הגשה: 12.01.2022

# תרגיל בית 2

## שאלה 1 – רשתות קונבולוציה

Data augmentation: גודל מדגם האימון: 200,000 תמונות. גודל מדגם המבחן: 10,000.

בניתי מדגם אימון חדש על סמך מדגם האימון המקורי של Cifar-10 המכיל 50,000 תמונות. עשיתי זאת באמצעות שכפול מדגם האימון ל4 מדגמי אימון חדשים, הפעלת אוגמנטציה שונה על כל אחד מהם ואיחודם לכדי מדגם אימון חדש בגודל 200,000 תמונות. להלן הפעולות שביצעתי על כל מדגם אימון, כאשר המשותף לכולם הוא הפיכת כל התמונות ל Tensors ונרמול לפי הממוצע וסטיית התקן של המדגם שהם

$mean = (0.4914, 0.4822, 0.4465),\ std = (0.2023, 0.1994, 0.2010)$.

**מדגם 1:** נירמלתי את הפיקסלים לפי הממוצע והחציון של כל שכבה שכתובים לעיל.

**מדגם 2:** הפעלתי על מדגם האימון את הפעולות הבאות:

```
transforms.RandomCrop(32, padding=4),
transforms.RandomHorizontalFlip(),
```

הפקודה הראשונה מוסיפה padding של 4 פיקסלים שחורים במיקומים רנדומליים מסביב לתמונה. בפועל לא מתבצע crop שכן התמונה כבר בגודל 32x32. הפקודה השנייה הופכת את התמונה אופקית בהסתברות 0.5.

**מדגם 3:** הפעלתי על מדגם האימון את הפעולה הבאה:

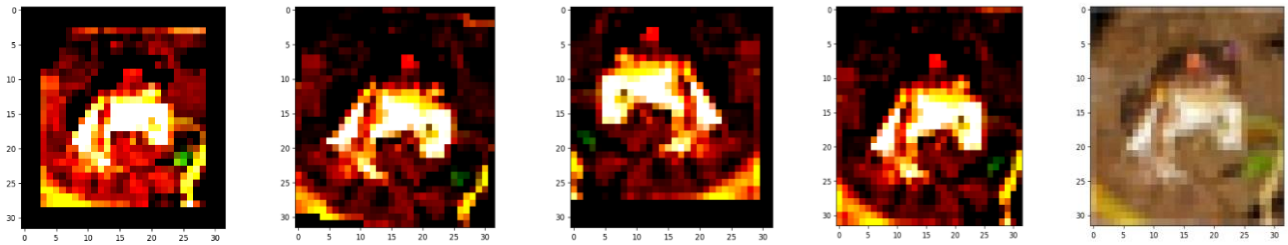
```
transforms.RandomRotation(10)
```

פעולה זו בוחרת זווית בצורה רנדומלית מהטווח (-10,10) ומסובבת את התמונה על פיה.

**מדגם 4:** הפעלתי על מדגם האימון את הפעולות הבאות:

```
transforms.RandomAffine(0, shear=10, scale=(0.8, 1.2)),
transforms.ColorJitter(brightness=0.2, contrast=0.2, saturation=0.2),
```

הפעולה הראשונה לא מסובבת את התמונה (0), מפעילה מתיחה\החלקה בזווית שבטווח (-10,10), ועושה scaling בערך שנע בין 0.8 ל 1.2. כל הערכים נבחרים באופן רנדומלי. הפעולה השנייה משנה את הצבעים של התמונה ומגדילה את הבהירות, את הניגודיות ואת החשיפה. להלן דוגמא לתמונה הראשונה מכל מדגם, בתוספת תמונה מהמדגם המקורי ללא העיבודים:



תמונה מקורית | מדגם 1 | מדגם 2 | מדגם 3 | מדגם 4

ארכיטקטורת המודל: 3 שכבות קונבולוציה עם 2 שכבות fully connected. מס' פרמטרים: 49,469.

**שכבת קונבולוציה 1:** 3 ערוצי קלט (RGB), 16 פילטרים, kernel בגודל (3,3) ו padding בגודל 1.

**שכבת max pooling**: ביצוע max pooling עם חלון בגודל 2X2.

**שכבת קונבולוציה 2**: 16 ערוצי קלט, 32 פילטרים, kernel בגודל (3,3) ו padding בגודל 1.

**שכבת max pooling**: ביצוע max pooling עם חלון בגודל 2X2.

**שכבת קונבולוציה 3:** 32 ערוצי קלט, 64 פילטרים kernel בגודל (3,3) ו padding בגודל 1.

**שכבת max pooling**: ביצוע max pooling עם חלון בגודל 2X2.

**שכבת Dropout:** הפיכה של 0.25 מהנוירונים בשכבה לאפס. הפעולה מונעת overfitting.

**שכבת fully connected 1:** קלט בגודל 64x4x4 ופלט בגודל 25. מקבלת את הפלט משכבת הקונבולוציה האחרונה שעבר הורדת מימד, שיטוח. הפלט של שכבה זו עובר אקטיבציה באמצעות Relu.

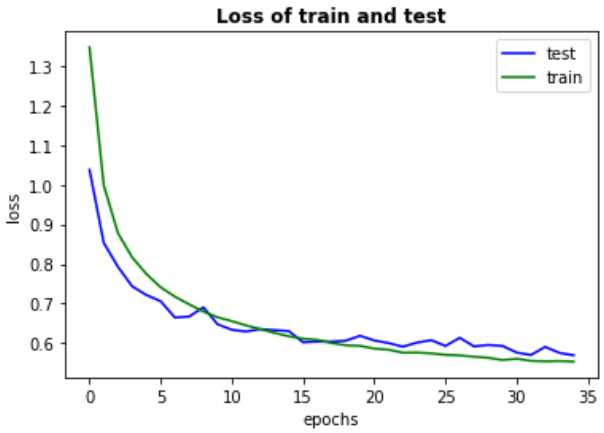
**שכבת fully connected 2:** קלט בגודל 25 ופלט בגודל 10 (כמות המחלקות).

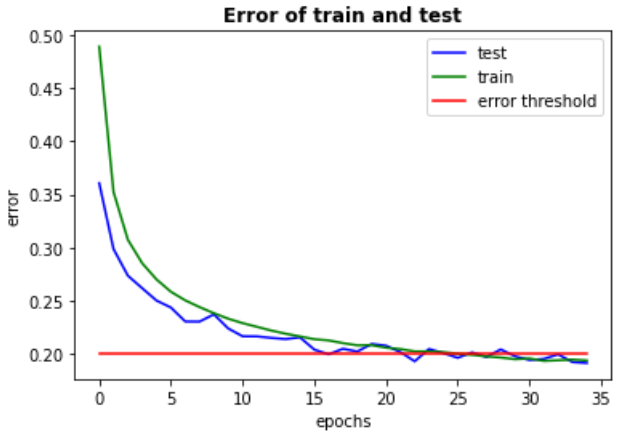
**פונקציית loss:** CrossEntropy.

אופטימיזציה: Adam optimizer עם learning rate = 0.001. היפר-פרמטרים: epochs = 36, batch size = 100.

פירוט הליך האימון: תחילה ניסיתי לבנות רשת עם מבנה קונבולוציה דומה לרשת הסופית שלי, אך עם קרנל בגודל 2x2 בשכבה האחרונה ולאחר מכן שכבת fully connected נוספת. בסך הכל 3 שכבות קונבולוציה ו3 שכבות fully connected. הדבר לא הניב הצלחה והגעתי ל55% דיוק. עם הרשת שכתובה כאן הגעתי לבערך 60% דיוק. החלטתי לשנות גישה ולנסות אוגמנטציות על הדאטה. בניסיון הראשון שלי (נרמול בלבד), הגעתי לכ65% דיוק אחרי הרבה epochs. ניסיתי עוד עיבודים דומים לאלו שכתובים מוקדם יותר בתרגיל אך דבר לא הצליח להרים את אחוזי הדיוק מעבר ל68%. החלטתי לבסוף להגדיל את מדגם האימון. בהתחלה לקחתי מדגם אימון אחד שהפעלתי עליו עיבודים כמו סיבוב ונרמול והכפלתי אותו פי 3. הדבר שיפר את אחוזי הדיוק בצורה גבוהה והצלחתי להגיע ל78% דיוק ואף 79% בחלק מהמקרים. לאחר תהייה על הנושא, הבנתי שהדבר הטוב ביותר לעשות הוא לשכפל את מדגם האימון פי 4, אך על כל חלק לבצע אוגמנטציות שונות. בכך אני אקבל ייצוגים שונים לתמונה והלומד יתפוס הרבה דברים בנוגע לאותה תמונה. לבסוף, הגעתי ל 80.83% דיוק על סט המבחן (כלומר error = 0.1917). תיעוד מהקוד:

```
Best test error: 0.1917
```



## שאלה 2 – Generative RNN

ייצוג הדאטה: נייצג כל אות כ one-hot vector בגודל כל האותיות שקיימות בדאטה. בנוסף, נייצג כל שם של שפה כ one-hot vector בגודל כל השפות שקיימות בדאטה. משום שהמודל הינו מודל גנרטיבי ומבוסס קונטקסט, הקלט למודל יהיה שרשור של: וקטור השפה + וקטור האות + וקטור ה hidden state.

עבור הקלט הראשוני, וקטור hidden state מאותחל כ וקטור אפסים בגודל שניתן כקלט לתוכנית (128).

מה אנו חוזים: המודל מקבל וקטורים של אות, שפה ו Hidden state, וחוזה את האות הבאה ברצף.

מתי אנו חוזים: בכל timestamp של קריאת רצף האותיות (מילה). אנו עוברים בצורה איטרטיבית על אותיות של מילה, מכניסים כל אות, שפת מקור ו hidden state קודם למודל ומקבלים את הפלט (האות הבאה רצף) ואת ה hidden state החדש. (*) כתוצאה מכך, ה loss מחושב בצורת סכום על פני כל האותיות ומוסיפים ל loss הכולל את ה loss הנוכחי מנורמל באורך המילה. להלן שרטוט שממחיש את ההליך שמתואר ואת מבנה הרשת:


ארכיטקטורת המודל:

**fully connected i2h:** קלט -> hidden state.

קלט: וקטור משורשר של הוקטורים: אות (גודל 58), שפה (גודל 18) ו hidden state (גודל 128). סך הכל וקטור בגודל 204.

פלט: וקטור hidden state בגודל 128.

**fully connected i2o:** קלט -> פלט ראשוני.

קלט: זהה ל i2h. פלט: וקטור output בגודל 58.

**Fully connected o2o:** פלט ראשוני + hidden state -> פלט סופי

קלט: וקטור output מהשכבה i2o משורשר לוקטור ה hidde state משכבת i2h. פלט: וקטור output בגודל 58.

**שכבת Dropout:** הפיכה של 0.1 מהנוירונים בoutput הסופי לאפס. הפעולה מונעת overfitting.

**שכבת softmax:** הפיכת וקטור ה output הסופי לוקטור עם הסתברויות (הסתברות לכל אות).

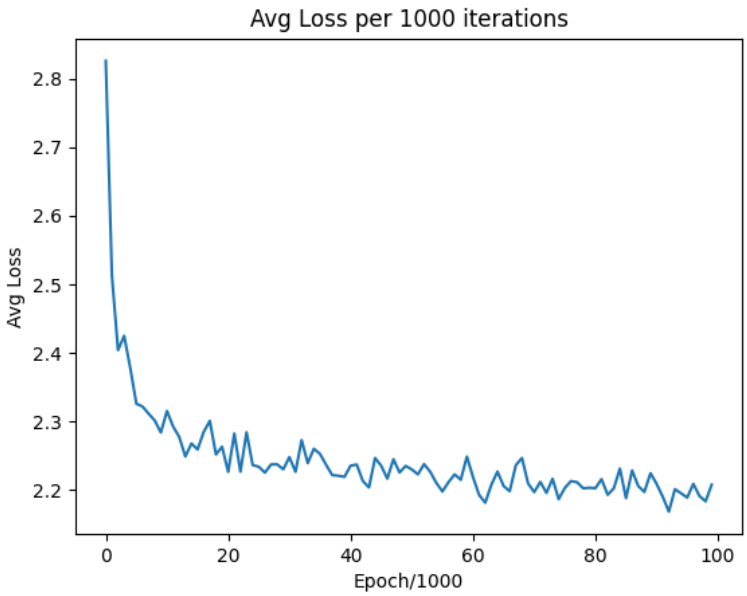
אופטימיזציה: Adam optimizer עם learning rate = 0.001.

היפר-פרמטרים: hidden size = 128, number of epochs = 100,000, plot_every = 1000

(כל כמה איטרציות לחשב loss ממוצע ולהוסיף אותו לרשימת הloss עבור הגרף).

הליך האימון: בכל epoch, מגרילים מילה ממדגם האימון עם הקטגוריה שלה. לאחר מכן, מחלצים את ה target (האות השנייה ואילך כולל תג EOS) ולבסוף הופכים את כולם ל one-hot vectors. עוברים בצורה איטרטיבית על האותיות ובכל פעם חוזים את האות הבאה. מחשבים את ה loss כמתואר טרם לכן (*) ובכל 1,000 איטרציות מוסיפים לרשימת ה loss את הממוצע על פני 1,000 האיטרציות.

להלן גרף עם ממוצעי ה loss בכל 1,000 איטרציות ו5 דוגמאות לפלטים:




```
Input Language: Russian, Input Letter: I, Output Name: Ivana
Input Language: Spanish, Input Letter: A, Output Name: Aberei
Input Language: Chinese, Input Letter: X, Output Name: Xien
Input Language: French, Input Letter: Z, Output Name: Zaucher
Input Language: German, Input Letter: G, Output Name: Gormer
```