למידה חישובית – תרגיל 3

מגישים: סהר בריבי, 311232730

אור אוקסנברג, 312460132

** התרגיל כולו (הקוד, הנתונים הסיכומים ומסמך זה) מופיע בGit בלינק:

https://github.com/oxenberg/systemEngineer/tree/master/ML/HW_3

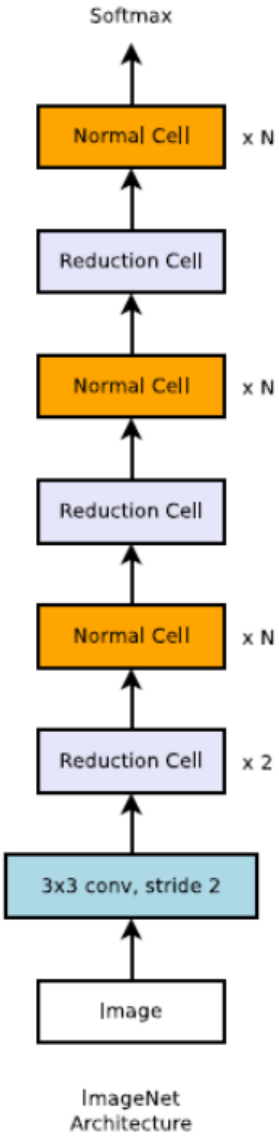
הרשת והשכבות:

לטובת התרגיל השתמשנו בשני מודלים שונים, NASNet-A וMobilenet-V2, על שניהם נפרט.

NASNet-A

NASNet הוא קיצור של Neural Architecture Search net, זהו מודל אשר אומן על דטסטים ידועים, CIFAR-10 או ImageNet, כאשר במקרה שלנו נעשה שימוש בImageNet. תהליך יצירת הרשת נעשה בצורה הבאה: בשלב הראשון, ניתן תנאי התחלה לרשת כאשר היא בנויה משני סוגי תאים: סוג אחד שנקרא Reduction Cell שמטרתו להוריד את כמות המימדים במהלך הזרימה ברשת, וסוג שני הנקרא Normal Cell, אשר שומר על המימדים של המידע. כמות החזרות של הNormal cells המסומנת בN בין כל Reduction cell, וכן כמות הפילטרים של קונבולציה התחלתית הם משתנים חופשיים שנקבעים מראש ונועדו לטובת סקיילביליות. בנוסף אליהם, יש לנו RNN אשר שולטת על המבנה הפנימי של כל תא. כל תא, מורכב ממספר בלוקים, אשר נבנים רקורסיבית בצורה הבאה:

1. בחירה 2 hidden states מתוך סט של hidden states
2. הפעלת אופרציה על כל אחד מהhidden states מתוך רשימת פעולות קיימת
3. הפעלת פעולה לשילוב הפלט הנוצר (לדוגמא חיבור), לטובת יצירת hidden state חדש.



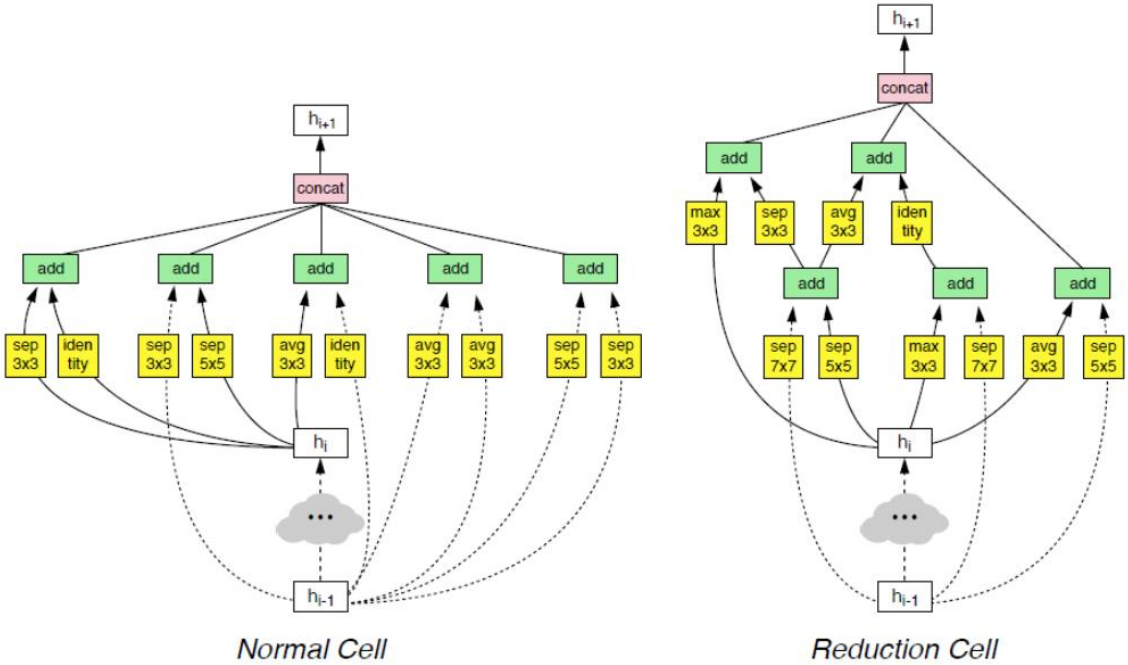
- identity
- 1x7 then 7x1 convolution
- 3x3 average pooling
- 5x5 max pooling
- 1x1 convolution
- 3x3 depthwise-separable conv
- 7x7 depthwise-separable conv
- 1x3 then 3x1 convolution
- 3x3 dilated convolution
- 3x3 max pooling
- 7x7 max pooling
- 3x3 convolution
- 5x5 depthwise-seperable conv

A set of operations to be selected

הRNN בנויה משכבת LSTM הבנויה מ100 יחידות נסתרות ו2*5B softmax פרדיקציות, כאשר B הוא מספר שנקבע מראש. ברשת שלנו, NASNet-A, B=5, כלומר יש 5 בלוקים, כפי שמוצג בתמונה (כל זוג ריבועים צהובים הוא בלוק).

בארכיטקטורה זו נעשה שימוש בשיטת רגולריזציה של ScheduledDropPath, אשר שיפרה בצורה משמעותית את ההכללה ברשתות אלו.

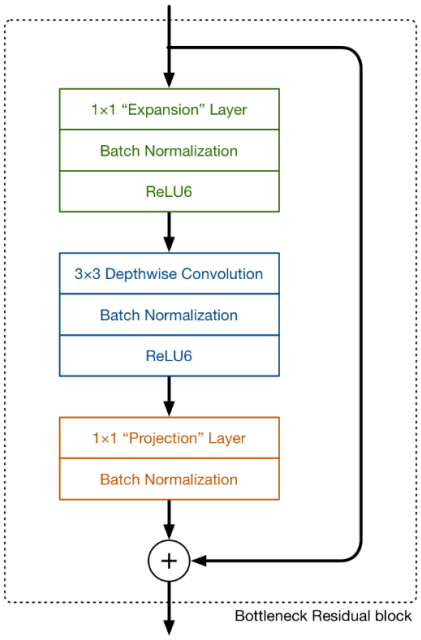
ברשת שאנחנו הפעלנו על דאטאסט התמונות, עשינו שימוש באימפלמנטציה של NASNet-Large מתוך NASNet-A שהשתמש ב18 תאים רגילים והתחיל עם 168 פילטרים של קונבולציה אחרי אימון על ImageNet. גודל התמונה באינפוט שהתקבל הוא 331*331 פיקסלים ולכן שינוי גודל התמונה שנעשה בתחילת התהליך תואם לגודל זה.



NASNet-A

MobileNet-V2

ארכיטקטורה זו יועדה בעיקר לעבודה על צ׳יפים ומכשירים ניידים בשל מהירותה. רשת זו בנויה מבלוקים, כאשר כל בלוק הוא בעל המרכיבים הבאים:



שכבות הExpansion layer וה-Projection layer הן שכבות המשנות את ממדי הדאטה, כאשר שינוי הממדים נקבע ע״י פרמטר הנקרא expansion factor. שכבת הExpansion מיועדת להרחבת הדאטה למען עיבוד מעמיק בשכבה האמצעית, שכבת הDepthwise. הProjection layer נועד להקטנת ממדי הדאטה לגודל המקורי. הסוד המסתתר מאחורי שיטה זו נובע ממתן הExpansion factor כהיפר-פרמטר. הרשת מבצעת פעולה הדומה לפתיחת קובץ zip במחשב, מבצעת עיבוד הדומה לקונבולוציה ולאחר מכן מכווצת אותו חזרה לגודלו המקורי. הרשת מתאימה את פרמטר הכיווץ בהתאם לדאטה המתקבל, בעזרת הווליד‍ציה. שכבת העיבוד העיקרית היא השכבה האמצעית, Depthwise layer, אשר בשונה משכבת קונבולוציה רגילה, מבצעת את חלון הקונבולוציה על ערוץ צבע אחד בלבד.

בנוסף, ניתן לראות כי לכל השכבות יש את נורמליזציה הBatch Normalization, ולכולן פרט לשכבת הProjection יש פונקציית אקטיבציה של ReLU6. לכל בלוק יש residual connection, שמטרתו לעזור לזרימת הgradient לאורך הרשת כלומר, לשפר את הלמידה, בתהליך הForward ו-Back Propagation.

מבנה הרשת הוא כפי שמתואר בטבלה,

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

כאשר bottleneck הוא הבלוק המתואר באיור הקודם. בטבלה, n היא כמות החזרות של אותה שכבה, c הוא כמות הchannelים של האאוטפוט, s הוא גודל הstride, t הוא הExpansion factor של כל שכבה. גודל הKernalים בכל שכבה הוא תמיד 3*3 כמתואר באיור הקודם.

הרשת בה השתמשנו מקבלת תמונה בגודל של 224*224 פיקסלים.

<u>תהליך הpreprocessing:</u>

תחילה סידרנו את התמונות בתיקיות, כאשר כל תיקייה כוללת פרחים מסוג מסוים. כלומר, כל תיקייה היא בעצם הlabel של התמונה. סידור זה מאפשר למודל לקבל את התמונות בצורה שכוללת כבר את הלייבל, לפי התיקייה שבה התמונה נמצאת.

תהליך נוסף הוא תהליך הdecoding של התמונה. התהליך הוא תהליך decode jpeg שמאפשר לנו לקחת תמונה שייבאנו בעזרת tensorflow, שמייבאת את התמונה, ועושה לו decoding מקידוד של jpeg לtensor.

במהלך תהליך זו אנחנו מגדירים כי התמונה היא בצבע ויש להשתמש בrgb.

מכאן, אנחנו לוקחים את הדאטה שנוצר מהתהליך ועושים לו המרה לtensor.float32. בנוסף, אנחנו מנרמלים את הערכים להיות בין 0 ל1.

השלב האחרון הוא שינוי של גודל התמונה, לפי הinput שהמודל שאנחנו משתמשים בו מצפה לקבל.

<u>נתונים סיכומים:</u>

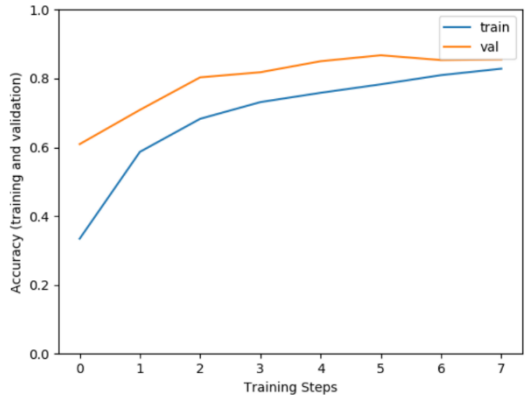
Model	Train	Validation	Test	Accuracy-Test	Accuracy-Validation	Accuracy-Train	Loss-Test	epoch
NASNet	70%	15%	15%	0.8726	0.8553	0.8286	1.4240	8
NASNet	30%	30%	40%	0.7771	0.8080	0.7204	1.7082	8
MobileNet	70%	15%	15%	0.941	0.9457	0.8909	1.2423	5
MobileNet	30%	30%	40%	0.8664	0.8742	0.8026	1.5016	5

גרפים:

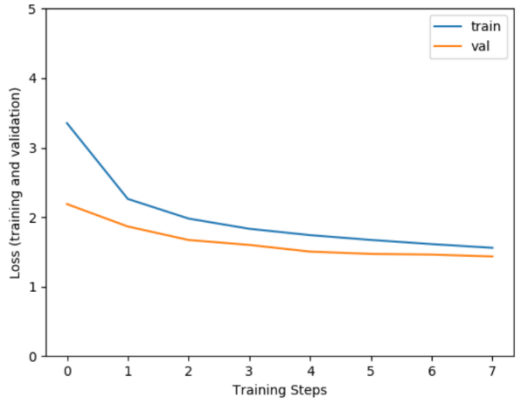
NASNet-A

חלוקה של 70%, 15%, 15% על Train, Validation, Test בהתאמה.

Accuracy:

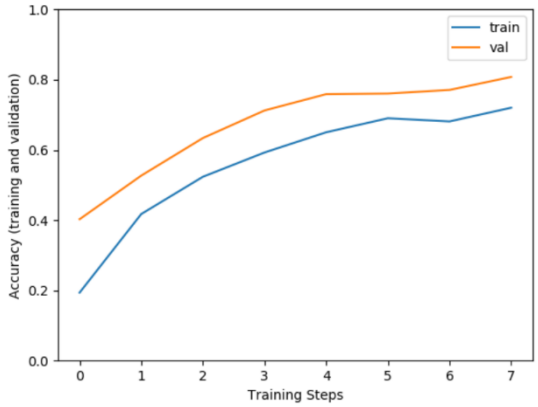


Cross Entropy:

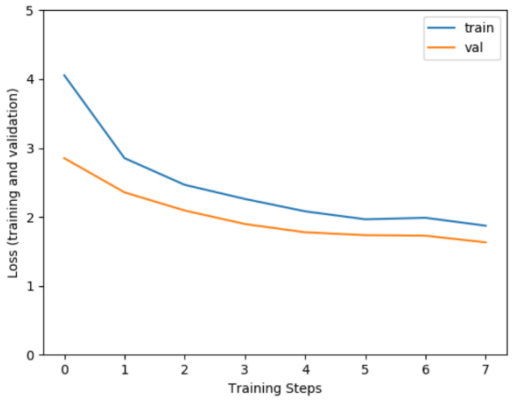


חלוקה של 30%, 30%, 40% על Train, Validation, Test בהתאמה.

Accuracy:



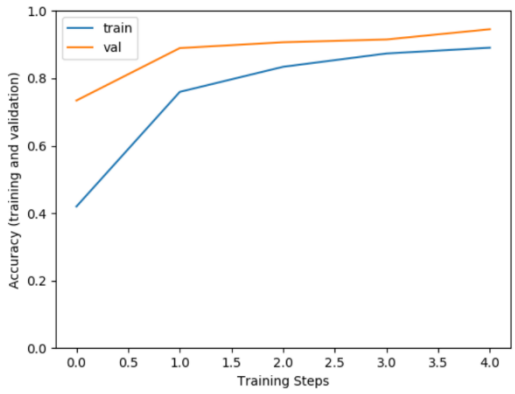
Cross Entropy:



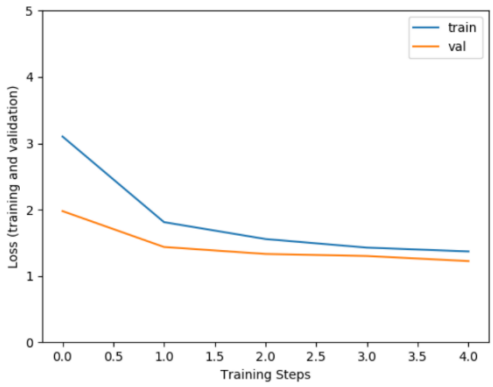
MobileNet-V2

חלוקה של 70%, 15%, 15% על Train, Validation, Test בהתאמה.

Accuracy:

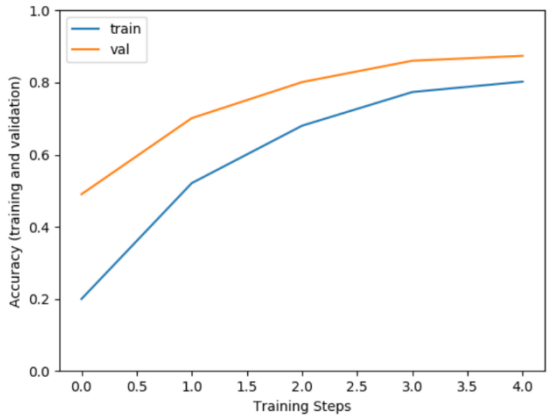


Cross Entropy:



חלוקה של 30%, 30%, 40% על Train, Validation, Test בהתאמה.

Accuracy :



Cross Entropy :