רשתות קונבולוציה מודרניות

רשתות קונבולוציה מצאו שימוש החל משנות התשעים: אחת הרשתות הפופולריות מתקופה זו היא LeNet-5, רשת אשר אומנה לזהות ספרות בכתב יד מתמונות 28X28 בעלות ערוץ צבע יחיד – תמונות בעלות אותם מימדים כאלו של אוסף הנתונים MNIST. רשת זו הראתה ביצועים שווי ערך לאלגוריתמי זיהוי תמונה אלטרנטיביים המשתמשים בחילוץ מאפיינים מהונדס מראש למטרה הנדרשת והזנת מאפיינים אלו לאלגוריתם למידה. זאת כאשר הקלט לרשת היה התמונות המקוריות, ללא כל עיבוד מקדים.

יחד עם זאת, הגידול העצום בפופולריות של רשתות קונבולוציה, ושל למידה עמוקה באופן כללי, התרחש עקב הביצועים יוצאי הדופן של הרשת AlexNet בתחרות ILSVRC-2012: תחרות זיהוי תמונה המבוססת על אוסף הנתונים הענק ImageNet. בפרק הנוכחי נסקור את הרכיבים אשר הובילו לנצחונה בתחרות, אלו הפכו סטנדרטיים בתכנון המבנה ותהליך האימון של רשתות נוירונים עמוקות כיום.

אוסף הנתונים ImageNet

בשנת 2009 פורסם לראשונה מסד הנתונים ImageNet המכיל מיליוני תמונות שנאספו ממנועי חיפוש ואתרי אינטרנט כגון Flickr. תמונות אלו סווגו על ידי משתמשים אנושיים למעל עשרים אלף קטגוריות שונות, חלקן חופפות (למשל כלב באופן כללי וכלב מגזע מסויים), ובהתאם תמונה נתונה סווגה בו זמנית למספר קטגוריות.

מתוך מסד זה בחרו מארגני התחרות ILSVRC אלף קטגוריות **זרות**, וסיפקו למשתתפי התחרות כאלף דגימות מכל קטגוריה, בסה"כ 1.2 מיליון תמונות לאימון. תחרויות פופולריות קודמות השתמשו בסט נתונים של עשרות אלפי תמונות השייכות לעשרות בודדות של קטגוריות ולכן זמינותו של סט נתונים בסדר גודל שכזה היוותה קרקע פורייה להצלחתן של רשתות נוירונים, אשר כזכור מצטיינות כאשר סט נתוני האימון גדול במיוחד.

גודל סט נתונים זה אינו מאפשר סיווג אנושי מדויק ומלא של כל התמונות: ייתכן למשל שקיימת באוסף נתונים תמונה השייכת לשתי קטגוריות בו זמנית, אך סווגה רק לאחת מהן. בכדי להתמודד עם אי דיוקים שכאלו, מדד ההצלחה הנבחר בתחרות היה דיוק של אחד מתוך 5 (Top-5 Accuracy), בו האלגוריתם הנמדד הציע חמש קטגוריות אפשריות עבור כל תמונה מסט הבדיקה. אם הקטגוריה הנכונה היא אחת מחמש אלו, נזקפה לזכות האלגוריתם הצלחה בסיווג התמונה הנתונה. במדד זה AlexNet הצליחה להגיע לאחוז סיווג נכון של 84.7%, בעוד שהאלגוריתם הבא בתור בתחרות זו, אשר השתמש בגישה קלאסית של חילוץ מאפיינים ומסווגים ליניאריים, הגיע לאחוז סיווג של 73.8% בלבד. בשנת 2015 אחוז ההצלחה של האלגוריתם המנצח (גם הוא רשת קונבולוציה, עליה נלמד בפרק הבא) כבר היה מעל ל-95%.

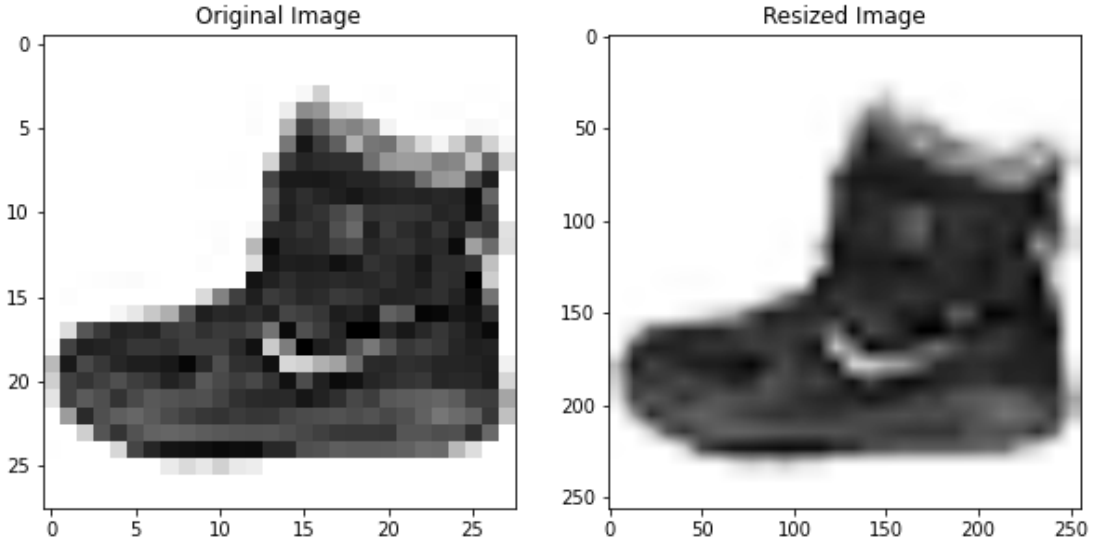
עיבוד מקדים והעשרת אוסף הנתונים

התמונות ב-ImageNet נאספו מרחבי האינטרנט ולכן הן לרוב מרזולוציה גדולה, גדולה מדי כקלט אפילו עבור רשת קונבולוציה. בהתאם, לפני הזנתן ברשת הן הוקטנו לגודל 256X256. בתהליך האימון תוצאה זו לא הוזנה ישירות אל הרשת אלא היוותה מקור להעשרת אוסף הנתונים: בעזרת טרנספורמציות פשוטות כגון שיקופים, סיבובים, שינויי צבע קלים וכו' ניתן לייצר מתמונה מקורית אחת אלפי תמונות נוספות מבלי לשנות את הקטגוריה אליה היא שייכת, בכך להגדיל את סט הנתונים באופן מלאכותי ולהימנע מהתאמת יתר.

עבור נותנים ויזואליים קיימת ב-PyTorch חבילת פונקציות נרחבת לעיבוד מקדים זה, ולפני שנמשיך את הדיון ב-AlexNet, נדגים את השימוש שלה להלן.

```
import torchvision.transforms as T
resize=T.Resize(256)
new_img=resize(img)
```

לאחר הרצת קטע הקוד הנ"ל, נקבל במשתנה `new_img` תמונה אשר הוגדלה/הוקטנה ל-256X256, ראו באיור להלן את השפעת הפונקציה על תמונה מהאוסף MNIST-Fashion.



לרוב נרצה להשתמש במספר פונקציות עיבוד מקדים, ולצורך זה ניתן לשרשרן כלהלן.

```
transforms=T.Compose([T.Resize(256),
             T.RandomCrop(224),
             T.RandomHorizontalFlip(0.5),
             T.RandomRotation(30),
             T.RandomPerspective(distortion_scale=0.3, p=0.5)])
new_img=transforms(img)
```

מספר תוצאות של הפעלת רצף טרנספורמציות זה, אשר מגדיל, חותך, משקף סביב האמצע, ומסובב בדרכים שונות ובאקראי את תמונת הקלט מאוייר להלן:

בכל התמונות ניכר כי זהו מגף ושימוש בסט אימון המועשר בתמונות אלו יניב רשת אשר למדה לזהות מגף גם דרך טרנספורמציות אלו, ובהכרח מותאמת פחות לסט הנתונים המקורי.

לבסוף, ניתן לשלב טרנספורמציות אלו ישירות בתהליך טעינת הנתונים בזכרון, על ידי הגדרתן בפרמטר `transform` של פונקציית הגדרת ה-Dataset. בצורה זו, הטרנספורמציות יופעלו פעם אחת על כל תמונה בעת טעינת כל minibatch, וכך הרשת תיחשף לתמונות "חדשות" בכל epoch.

נקודה עדינה שדורשת התייחסות כאשר אנו מעשירים את סט הנתונים היא שאת סט הבדיקה יש להעביר אל הרשת דרך טרנספורמציות דומות, בכדי להתאימו לקלט עליו הרשת אומנה. יחד עם זאת, ברצוננו להימנע מהפעלת טרנספורמציות אקראיות על סט הבדיקה, דבר אשר יהפוך את החיזוי לתלוי בהגרלת המשתנים המקריים. אם כן, סדרת הטרנספורמציות עבור סט הבדיקה, המתאימה לסט אימון שהועשר כנ"ל מופיעה בקטע הקוד הבא, ואותה יש להעביר להגדרת אובייקט ה-Dataset של סט הבדיקה. ראו כי הגזירה הרנדומלית הוחלפה בערך הממוצע שלה: גזירה במרכז התמונה.

```
test_transforms=T.Compose([T.ToTensor(),
                           T.Resize(256),
                           T.CenterCrop(224)])
```

ארכיטקטורת הרשת AlexNet

בקטע הקוד הבא מופיעה ארכיטקטורת הרשת AlexNet, כמחלקה של PyTorch. שימו לב שזו הפעם הראשונה בה אנו מגדירים רשת שלמה כמחלקה היורשת מ-`nn.Module`, אך אין הדבר שונה מהגדרת שכבה יחידה בצורה זו.

```
class AlexNet(nn.Module):
    def __init__(self, dropout=0.5):
        super().__init__()
        self.features = nn.Sequential(
            nn.Conv2d(3, 64, 11, stride=4, padding=2),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=3, stride=2),
            nn.Conv2d(64, 192, 5, padding=2),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=3, stride=2),
            nn.Conv2d(192, 384, 3, padding=1),
            nn.ReLU(),
            nn.Conv2d(384, 256, 3, padding=1),
            nn.ReLU(),
            nn.Conv2d(256, 256, 3, padding=1),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=3, stride=2))
        self.classifier = nn.Sequential(
            nn.Dropout(p=dropout),
            nn.Linear(256 * 6 * 6, 4096),
            nn.ReLU(),
            nn.Dropout(p=dropout),
            nn.Linear(4096, 4096),
            nn.ReLU(),
            nn.Linear(4096, 1000))
    def forward(self, x):
        x = self.features(x)
        x = self.classifier(x.flatten(start_dim=1))
        return x
```

מעיון בארכיטקטורה עולים מספר פרטים מעניינים. ראשית, נשים לב שפלט הרשת הוא שכבה ליניארית – אלו הציונים הגולמיים אשר יש להעביר לפונקציית ה-Softmax ולקבל את הסתברויות השייכות לכל אחת מאלף המחלקות. כזכור, בעזרת הסתברויות אלו אנו מחשבים את פונקציית המחיר המתאימה למשימת סיווג, היא האנטרופיה הצולבת. ב-PyTorch ניתן להעביר ציונים אלו ישירות לפונקציית המחיר המבצעת חישוב זה בצורה יעילה. על כן, על מנת לאמן רשת זו יש להגדיר את פונקציית המחיר כך:

```
CE_loss = nn.CrossEntropyLoss()
```

ולא בעזרת הפונקציה `nn.NLLLoss`, כפי שעשינו בעבר כאשר בסוף הרשת היתה שכבת `nn.LogSoftmax`.

שנית, גרעין הקונבולוציה הגדול בשכבה הראשונה, והשימוש בשכבות איגום מפחיתים את גודל הקלט ל-13X13 באמצע הרשת ואף ל-6X6 לאחר שכבת האיגום האחרונה. יחד עם זאת, מספר הערוצים גדול עד לערך מקסימלי של 384. הרעיון מאחורי טרנספורמציות אלו הוא לאפשר לרשת ללמוד מספר רב של מחלצי מאפיינים המתייחסים לחלקים נרחבים ברשת, שהרי עקב הפחתת המימד פיקסל יחיד במאפיין מאמצע הרשת מחושב על בסיס מספר רב של פיקסלים בתמונה המקורית.

פרט שלישי מעניין הוא השימוש בשכבות Dropout ופונקציית האקטיבציה ReLU, דברים שבשגרה בשימוש המודרני, אך היו בתחילת דרכם בעת פרסום המאמר, ואין ספק שהצלחתה של AlexNet תרמה לפופולריות של רכיבים אלו.

לבסוף, ניתן לראות שהחישוב ברשת מחולק לשניים: רשת קונבולוציה המחלצת מאפיינים ומעבירה אותם למסווג בעל קישוריות מלאה סטנדרטי – זהו "ראש הסיווג" אשר ניתן להחלפה אם ברצוננו לבצע משימה אחרת, כגון רגרסיה, על בסיס רשת זו.

שני פרטים חשובים אשר לא באים לידי ביטוי במימוש הנ"ל, אך היו אינטגרליים להצלחתה הם השימוש בנרמול, בדומה ל-Batch Normalization, על בסיס מימד הערוץ במקום מימד ה-batch וכן אימון הרשת על גבי מאיץ גרפי. גם תוך שימוש בשני מאיצים במקביל – נדרש כשבוע של זמן ריצה עבור אימון הרשת המקורית.

שאלות לתרגול

1.
א. הגדירו מחלקה היורשת מ-`nn.Module` ובה ממומשת ארכיטקטורת הרשת LeNet-5 (מצאו איור של ארכיטקטורה זו במאמר המקורי הנמצא באתר הקורס). ניתן להחליף רכיבים כגון פונקציית האקטיבציה tanh באלטרנטיבות מודרניות כגון ReLU.
ב. אמנו את הרשת לזהות את פרטי הלבוש מאוסף הנתונים Fashion-MNIST, עד אשר מתקבלת התאמת יתר ברורה.
ג. בצעו לפני הזנת התמונות ברשת טרנספורמציות שונות על תמונות הקלט למטרת העשרת אוסף הנתונים ואמנו את הרשת שוב. האם התקבלה התאמת יתר?
2. העבירו דרך הרשת AlexNet דגימה אקראית של קלט במימדים הצפויים (תמונות צבע 3X224X224), תוך כדי הדפסת גודל הטנזור לאחר מעבר בכל שכבה.
רמז: השתמשו במתודה `module.children()` המחזירה גנרטור של תתי השכבות במודול.
3. מהו מספר הפרמטרים ברשת AlexNet?
הנחיות:
- מצאו את גודל גרעין הקונבולוציה עבור כל שכבה, תוך התחשבות במספר ערוצי הקלט והפלט. זכרו שלכל ערוץ פלט יש גם פרמטר bias.
- זכרו שבשכבה ליניארית כל נוירון פלט מחובר לכל נוירון קלט, ולכל חיבור כזה קיים פרמטר.
- תוכלו לעשות זאת באופן אוטומטי על ידי מעבר על כל השכבות ברשת וצבירת הגודל של הפרמטרים שלהן.
4. על סמך התשובה לשאלה הקודמת, האם תוכלו לשער מדוע בארכיטקטורת AlexNet השתמשו בשכבות Dropout במסווג בלבד?