פרויקט גמר 5 יחידות לימוד-
למידת מכונות Deep Learning

שם בית הספר: מקיף יא הראשונים

לוגו בית הספר:

שם התלמיד: מיה עזרא

תעודת זהות התלמיד: 328151899

שם העבודה: זיהוי תמונות

שם המנחה: דינה קראוס

תאריך ההגשה: 6.6.2024

תוכן עניינים

מבוא

במסגרת פרויקט 5 היחידות במגמה, התבקשנו לבצע פרויקט ב machine learning, פרויקט הקשור לתחום deep learning. ארחיב מעט על התחום, ולאחר מכן אפרט על מטרת הפרויקט, אופן הביצוע, קשיים ושאיפות.

רקע- Deep learning

Deep learning (למידה עמוקה) הוא תחום של למידת מכונה, המשתמש במבנים של רשתות נוירונים מלאכותיות בהשראת מבנה ותפקוד המוח האנושי. רשתות נוירונים עמוקות מורכבות משכבות רבות של נוירונים מלאכותיים, הנקראים שכבות מוסתרות, המסוגלות ללמוד ולהבין ייצוגים מורכבים של נתונים. בתהליך הלמידה, הרשת מקבלת נתונים בקלט, מעבירה אותם דרך השכבות המוסתרות, ומסיקה תוצאה בקצה השני. כל שכבה לומדת תכונות שונות של הנתונים, כאשר השכבות הראשונות לומדות תכונות פשוטות (כגון קווים וזוויות בתמונות), והשכבות העמוקות יותר לומדות תכונות מורכבות יותר (כגון זיהוי אובייקטים שלמים).
אחד היתרונות המשמעותיים של למידה עמוקה הוא היכולת שלה להתמודד עם כמויות גדולות של נתונים ולהפיק תובנות מדויקות מהם, מה שמוביל לפריצות דרך במגוון תחומים, כולל עיבוד תמונה, זיהוי קול, תרגום שפות, ועוד.
למידה עמוקה זכתה לפופולריות רבה בשנים האחרונות בזכות השיפורים הטכנולוגיים בכוח עיבוד המחשבים וזמינות כמויות גדולות של נתונים, אשר מאפשרים לאמן רשתות נוירונים מורכבות ועמוקות יותר.

רקע לפרויקט

<u>מדוע בחרתי בפרויקט זה? -</u> כאשר התבקשתי לבחור נושא לפרויקט שלי בלמידת מכונות, ידעתי שאני רוצה לבחור בפרויקט שיהיה בתחום חדשני וגם פרויקט שיצור עבורי אתגר כלשהו, כי אני בן אדם שאוהב אתגרים. זיהוי אובייקטים הינו נושא חדשני ועדכני, שתמיד יכול לתרום ולסייע לאנשים ולחברה וזו אחת מהסיבות למה בחרתי בפרויקט- אני מאמינה שנושא זה יכול לסייע לתחומים שונים באבטחה, רפואה ועוד. אני, מאוד מתעניינת בטכנולוגיה ובמה שהיא יכולה לעשות ואני אוהבת ללמוד דברים חדשים, ופרויקט זה נתן לי הזדמנות להתנסות בטכנולוגיות מתקדמות כמו TensorFlow ו-Keras.

<u>מטרת הפרויקט -</u> בפרויקט זה החלטתי ליצור מודל שיזהה עבורי איזה סוג תמונה מתוך data של תמונות מספרים ותמונות שאינן מספרים (חפצים, בעלי חיים...), כלומר המטרה היא שכאשר יקבל הפרויקט שלי תמונות של מספרים- יוכל לזהות כל מספר בנפרד ולהראות זאת למשתמש ודומה עבור תמונות שאינן מספרים – יוכל לזהות האם מדובר במטוס או צפרדע ובקיצור - מטרת הפרויקט היא לפתח מערכת לזיהוי אובייקטים בתמונות באמצעות למידה עמוקה. המערכת תוכל לזהות ולסווג אובייקטים שונים בתמונות באופן אוטומטי.

<u>אופן הפעולה העתידי של הפרויקט -</u> כאשר חשבתי על הפרויקט וכיצד אעשה אותו, הייתי צריכה לחשוב על אופן הפעולה העתידי שרציתי שיהיה בפרויקט, כלומר כיצד תתבצע מטרת הפרויקט. כאשר חשבתי על אופן הפעולה, ידעתי שחשוב לי לשמור וליישם מספר דגשים: חוויות משתמש ידידותית (למשתמש יהיה קל להשתמש במערכת בקלות לא משנה רמת הידע הטכני שלו, תהליך העלאת התמונה וקבלת התוצאה פשוט וברור.), המודל יהיה יעיל ומדויק, הפרויקט יביא תוצאות בזמן אמת, הפרויקט יהיה בנוי בצורה גמישה כך שיהיה ניתן

לשדרג להתאים ולשפר את המודל. אופן הפעולה יהיה כך: המערכת תקבל תמונות כקלט, תעביר אותן דרך מודל למידה עמוקה מאומן, ותסווג את האובייקטים המופיעים בתמונות לקטגוריות מוגדרות מראש. התוצאות יופיעו למשתמש בצורה ברורה.

קהל היעד – קהל היעד שלי התחבר לי לסיבה מדוע בפרויקט זה, כד לתרום ולסייע לחברה. חשבתי מי יכול להרוויח ולמי יכול לתרום פרויקט זיהוי אובייקטים, ועלה לי לראש אבטחה, רפואה, בטיחות, וכן בכל יישום בו צריך לזהות אובייקטים.

תהליך המחקר

1. בשוק, יש מספר חברות מובילות בשוק שעוסקות בזיהוי אובייקטים באמצעות deep learning, חברות כגון Google Cloud Vision AI , Amazon Rekognition, Azure Computer Vision, IBM Watson Visual Recognition, Clarifai ועוד.
2. הפרויקט שלי שונה וחדשני מהפלטפורמות הגדולות. הוא מותאם אישית לצרכים ספציפיים, מאפשר שימוש קל ונוח, ומשתמש בטכנולוגיות חדשות ומתקדמות כמו TensorFlow ו- Keras. באמצעות טכנולוגיות אלו, ניתן להשיג ביצועים גבוהים ודיוק מרבי בזיהוי האובייקטים. בנוסף, הפרויקט פתוח לקהילה ומאפשר התאמות ושיפורים על ידי מפתחים אחרים, מה שמבטיח עדכניות ושיפור מתמיד.
3. בפרויקט השתמשתי במאמרים אקדמיים, תיעוד טכנולוגי וספרות מקצועית כדי להבין את היסודות ואת הטכנולוגיות השונות בתחום זיהוי האובייקטים. מקורות אלו סיפקו לי את הידע הבסיסי ואת הגישות המתקדמות ביותר בתחום Deep learning.

אתגרים ופתרונות בפרויקט

תחום ה deep learning -הינו תחום חדשני מאוד, והינו תחום שנמצא בפיתוח תמידי – כך שאני יכולה להיתקל בבעיות בפרויקט כאשר דברים יתחדשו בזמן שאעבוד על הפרויקט.
בפרויקט שלי אני אעמוד מול מגוון רחב של אובייקטים מסוגים שונים ולבחור ולסנן אותם בדיוק, לוודא שהמודל מזהה את כל הקטגוריות הרלוונטיות. פתרון לכך יכול להיות לאסוף המון נתונים מכל מני מאגרים שונים ולהתאים אל הצרכים של המודל.
בעיה נוספת יכולה לאחסן ולנהל כמות גדולה של data ולנהל בצורה יעילה את התמונות. פתרון לכך יכול להיות להשתמש בשירותי ענן לאחסון נתונים.

הפרויקט בא לענות על הצורך הגובר במערכות אוטומטיות לזיהוי אובייקטים. הצורך הזה נובע מהדרישה לשפר את הבטיחות, אבטחה והפיכת תהליכים שונים לאוטומטיים ויעילים יותר.

מבנה / ארכיטקטורה

את המבנה של הפרויקט אחלק לשלושה חלקים: החלק הראשון תיעוד שלב האיסוף, חקר הנתונים והכנתם לאימון. החלק השני יהיה שלב בניית המודל ותהליך האימון והחלק השלישי יהיה שלב היישום.

איסוף, הכנה וניתוח הנתונים(Collect, Prepare, and Analyze Data)

תיאור מבנה הנתונים (Dataset) :

בפרויקט שלי החלטתי להשתמש בשני סטים של דאטה- שני סוגים של תמונות. החלטתי לעשות זאת על מנת לוודא את יעילות הפרויקט ולוודא שהפרויקט עושה ומבצע את מטרתו לא רק עבור תמונות ספציפיות- לקחתי סט תמונות של מספרים וסט תמונות של אובייקטים רנדומליים (בעלי חיים, חפצים וכו) כך בעצם ווידאתי שהפרויקט מבצע את מטרתו לא רק עבור מספרים ולא רק עבור אובייקטים.

התחלתי לחפש דאטה וידעתי שאני צריכה כמות גדולה של תמונות על מנת לבצע את הפרויקט. פניתי לאתר TensorFlow. לא ידעתי בהכרח איזה שני סוגי תמונות אשתמש אבל ידעתי שאני רוצה שני סוגים. בקישור זה
https://www.tensorflow.org/api_docs/python/tf/keras/datasets
מצאתי את שני סוגי הדאטה בהם אני אשתמש- cifar10 ו mnist – דאטה של אובייקטים ושל תמונות.

-cifar10
https://www.tensorflow.org/api_docs/python/tf/keras/datasets/cifar10/load_data

https://www.cs.toronto.edu/%7Ekriz/cifar.html

https://www.cs.toronto.edu/%7Ekriz/cifar.html

-mnist

https://www.tensorflow.org/api_docs/python/tf/keras/datasets/mnist/load_data

/http://yann.lecun.com/exdb/mnist

/http://yann.lecun.com/exdb/mnist

תיאור וניתוח הנתונים הגולמיים:

הנתונים כוללים תמונות בגדלים ובפורמטים שונים. יש צורך בתהליך של נרמול ותיקון כדי להביא את כל התמונות לפורמט אחיד המתאים לאימון המודל

הגדלים והפורמטים השונים הם:

תיאור תהליך הכנת הנתונים לאימון.:

הנתונים חולקו לקבוצות אימון ובדיקה. תהליך הכנת הנתונים כלל מספר שלבים:

1. הבאת הנתונים מן האתר:

```
from tensorflow.keras.datasets import mnist
```

```
from tensorflow.keras.datasets import cifar10
```

```
(train_images, train_labels), (test_images, test_labels) =
mnist.load_data()
    (train_images, train_labels), (test_images, test_labels) =
cifar10.load_data()
```

למשתמש יש אפשרות לבחור על איזה סוג תמונות (דאטה) הוא רוצה לבצע את הרצת הפרויקט.

2. שינוי גודל כל התמונות ל28x28 פיקסלים. תהליך זה קורה כמה פעמים במהלך הקוד. שלב זה הכרחי על מנת הרצת המודל על התמונות.
3. שינוי צבע כל התמונות לשחור לבן.
4. נרמול הערכים לפורמט המתאים למודל. תהליך הנרמול חשוב כדי להביא את ערכי הפיקסלים לטווח אחיד של 0 עד 1, מה שמשפר את ביצועי המודל ומאיץ את תהליך האימון.
שורת הקוד:

```
 img_array = np.array(img)
```

היא השורה שמבצעת את הנרמול לפורמט המתאים למודל על ידי חלוקת ערכי הפיקסלים ב-255. נמצאת בקובץ tools.py

בנייה ואימון המודל(Build and Train Deep Learning Model)

תיאור גרפי של המודל עליו בוצע האימון:

המודל בנוי מרשת נוירונים עמוקה הכוללת שכבות שונות עליהן ארחיב בהמשך.

בפרויקט שלי השתמשתי במודלים encoder + decoder. ארחיב עליהם.

שימוש במבנה של Encoder-Decoder בפרויקט שלי מאפשר למודל למצות תכונות מורכבות מהתמונות ולשחזר את המידע בצורה מדויקת. זה עוזר לשפר את דיוק המודל ולמקסם את היכולת שלו לבצע זיהוי אובייקטים מורכב ומדויק. השימוש במבנה זה מאפשר

להתמקד בתכונות החשובות ביותר בתמונה ולהבטיח שהמודל יבצע סיווג מדויק של האובייקטים.

Encoder

ה-encoder הוא מרכיב חשוב במודלי למידת מכונה, במיוחד ברשתות עצביות עמוקות ובמערכות של למידת ייצוג. תפקידו לעבד ולהפחית את מימדיות הקלט לייצוג קומפקטי ומופשט. ה-encoder מקבל את הנתונים הגולמיים, מעביר אותם דרך מספר שכבות נוירונים שמקטינות בהדרגה את מימדיו, ובסופו של תהליך יוצר ייצוג חבוי שמכיל פחות נתונים אך שומר על המידע החשוב. ה-encoder משמש לדחיסת מידע, הורדת מימדים וייצוג תכונות חשובות, מה שמקל על עיבוד נוסף של הנתונים או שימושים כמו דחיסת נתונים ושחזורם באמצעות decoder.

ובקצרה- ה Encoder-לוקח תמונה וממיר אותה למרחב תכונות עשיר ומכווץ שמייצג את המידע החשוב בתמונה בצורה קומפקטית

.

```
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 28, 28, 1)]       0

conv2d_23 (Conv2D)           (None, 28, 28, 32)        320

batch_normalization_22 (Bat  (None, 28, 28, 32)        128
chNormalization)

conv2d_24 (Conv2D)           (None, 28, 28, 32)        9248

batch_normalization_23 (Bat  (None, 28, 28, 32)        128
chNormalization)

max_pooling2d_4 (MaxPooling  (None, 14, 14, 32)        0
2D)

conv2d_25 (Conv2D)           (None, 14, 14, 64)        18496

batch_normalization_24 (Bat  (None, 14, 14, 64)        256
chNormalization)

conv2d_26 (Conv2D)           (None, 14, 14, 64)        36928

batch_normalization_25 (Bat  (None, 14, 14, 64)        256
chNormalization)

max_pooling2d_5 (MaxPooling  (None, 7, 7, 64)          0
2D)

conv2d_27 (Conv2D)           (None, 7, 7, 128)         73856

batch_normalization_26 (Bat  (None, 7, 7, 128)         512
chNormalization)

conv2d_28 (Conv2D)           (None, 7, 7, 128)         147584

batch_normalization_27 (Bat  (None, 7, 7, 128)         512
chNormalization)

conv2d_29 (Conv2D)           (None, 7, 7, 256)         295168

batch_normalization_28 (Bat  (None, 7, 7, 256)         1024
chNormalization)

conv2d_30 (Conv2D)           (None, 7, 7, 256)         590080

batch_normalization_29 (Bat  (None, 7, 7, 256)         1024
chNormalization)

conv2d_31 (Conv2D)           (None, 7, 7, 128)         295040

batch_normalization_30 (Bat  (None, 7, 7, 128)         512
chNormalization)
```

```
conv2d_33 (Conv2D)          (None, 7, 7, 64)          73792

batch_normalization_32 (Bat  (None, 7, 7, 64)         256
chNormalization)

conv2d_34 (Conv2D)          (None, 7, 7, 64)          36928

batch_normalization_33 (Bat  (None, 7, 7, 64)         256
chNormalization)

up_sampling2d_2 (UpSampling  (None, 14, 14, 64)       0
2D)

conv2d_35 (Conv2D)          (None, 14, 14, 32)        18464

batch_normalization_34 (Bat  (None, 14, 14, 32)       128
chNormalization)

conv2d_36 (Conv2D)          (None, 14, 14, 32)        9248

batch_normalization_35 (Bat  (None, 14, 14, 32)       128
chNormalization)

up_sampling2d_3 (UpSampling  (None, 28, 28, 32)       0
2D)

conv2d_37 (Conv2D)          (None, 28, 28, 1)         289

=================================================================
Total params: 1,758,657
Trainable params: 1,755,841
Non-trainable params: 2,816
_________________________________________________________________
Epoch 1/5
Traceback (most recent call last):
```

Decoder

ה decoder-הוא מרכיב חשוב במודלי למידת מכונה, במיוחד באדריכלות של autoencodersורשתות עצביות עמוקות. תפקידו לקחת את הייצוג הקומפקטי והמופשט שנוצר על ידי ה encoder-ולהמיר אותו בחזרה לצורתו המקורית או לצורה דומה לה. ה-decoderמקבל את הקוד או הייצוג החבוי ומעביר אותו דרך מספר שכבות נוירונים, כל שכבה מגדילה בהדרגה את המימדים של הקלט, עד שהמידע משוחזר לצורתו הראשונית. ה decoder-משמש לשחזור מידע, יצירת נתונים חדשים מבוססי ייצוגים חבויים, ושיפור הדיוק של דגמים על ידי עיבוד חוזר של הנתונים. הוא חלק קריטי במערכות כמו autoencoders, Generative Adversarial Networks (GANs) ורשתות תרגום מכונה.

ובקצרה- חלק ה Decoder-משמש לשחזור המידע המכווץ למידע מלא. הוא מבצע הרחבה של מרחב התכונות לכדי תמונה או מבנה מלא של נתונים.

```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 28, 28, 1)]       0

conv2d_15 (Conv2D)           (None, 28, 28, 32)        320

batch_normalization_14 (Bat  (None, 28, 28, 32)        128
chNormalization)

conv2d_16 (Conv2D)           (None, 28, 28, 32)        9248

batch_normalization_15 (Bat  (None, 28, 28, 32)        128
chNormalization)

max_pooling2d_2 (MaxPooling  (None, 14, 14, 32)        0
2D)

conv2d_17 (Conv2D)           (None, 14, 14, 64)        18496

batch_normalization_16 (Bat  (None, 14, 14, 64)        256
chNormalization)

conv2d_18 (Conv2D)           (None, 14, 14, 64)        36928

batch_normalization_17 (Bat  (None, 14, 14, 64)        256
chNormalization)

max_pooling2d_3 (MaxPooling  (None, 7, 7, 64)          0
2D)

conv2d_19 (Conv2D)           (None, 7, 7, 128)         73856

batch_normalization_18 (Bat  (None, 7, 7, 128)         512
chNormalization)

conv2d_20 (Conv2D)           (None, 7, 7, 128)         147584

batch_normalization_19 (Bat  (None, 7, 7, 128)         512
chNormalization)

conv2d_21 (Conv2D)           (None, 7, 7, 256)         295168

batch_normalization_20 (Bat  (None, 7, 7, 256)         1024
chNormalization)

conv2d_22 (Conv2D)           (None, 7, 7, 256)         590080

batch_normalization_21 (Bat  (None, 7, 7, 256)         1024
chNormalization)

flatten (Flatten)            (None, 12544)             0

dense (Dense)                (None, 128)               1605760

dense_1 (Dense)              (None, 10)                1290

=================================================================
Total params: 2,782,570
Trainable params: 1,607,050
Non-trainable params: 1,175,520
```

הסבר על סוגי השכבות השונים ברשת:

במודל יש מספר שכבות שונות שהן:
input,flatten,conv2D,Dense,Maxpooling2D,batch_normalization, UpSampling2D

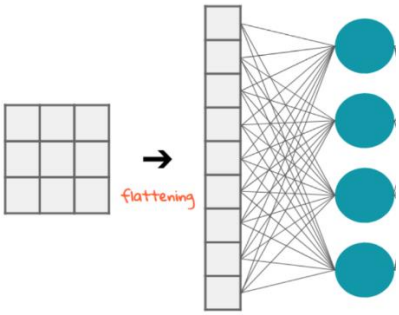
אסביר כעת על כל שכבה ושכבה.

Input layer

שכבת הקלט (Input Layer) היא השכבה הראשונה ברשת עצבית מלאכותית, והיא אחראית על קבלת הנתונים הגולמיים שמוזנים למערכת. שכבת הקלט אינה מבצעת עיבוד על הנתונים אלא רק מעבירה אותם לשכבות הבאות ברשת לצורך עיבוד נוסף.

Flatten layer

השכבה Flatten (שיטוח) ברשת עצבית משמשת להמרת נתונים מרובי מימדים לווקטור חד-ממדי של ערכים. תפקידה הוא לשטח נתונים כמו תמונות (בעלות מימדים של גובה, רוחב וערוצים) לווקטור שמכיל את כל הפיקסלים ברצף, כדי שניתן יהיה להזין אותם לשכבות Fully Connected (שכבות צפופות) ברשת. השכבה מקבלת נתונים במימדים מרובים וממירה אותם לווקטור חד-ממדי. הפיכת מטריצה לוקטור.



Conv2D

השכבה Conv2D היא שכבה ברשתות עצביות קונבולוציוניות (CNN) המשמשת לעיבוד ועיבוד תמונות ונתונים מרחביים. תפקידה הוא להפעיל מסננים (kernels) על הקלט כדי לזהות תכונות מקומיות כמו קצוות, זוויות, ודפוסים מורכבים יותר. כל מסנן מחליק על התמונה ומבצע פעולת קונבולוציה, שבה הוא מכפיל את ערכי הפיקסלים המקומיים ומשקלל אותם לסכום יחיד, מה שיוצר מפות תכונה (feature maps) שמייצגות את התכונות שנלמדו. השכבה Conv2D מאפשרת לרשת לזהות תכונות מורכבות בהדרגה דרך מספר שכבות קונבולוציוניות, מה שמוביל לשיפור בביצועים במשימות כמו זיהוי תמונות, סיווג, וזיהוי אובייקטים.

Dense

השכבה Dense, המכונה גם Fully Connected, היא שכבה ברשתות עצביות שבה כל נוירון מחובר לכל נוירון בשכבה הקודמת. כל חיבור בין נוירונים כולל משקל שנלמד במהלך תהליך האימון. תפקידה של השכבה Dense הוא לבצע שילוב ליניארי של התכונות שהופקו

מהשכבות הקודמות ולאחר מכן להחיל פונקציית הפעלה כדי להוסיף אי-ליניאריות, מה שמאפשר לרשת ללמוד ולייצג תבניות מורכבות יותר. השכבה משמשת בעיקר בסוף הרשת כדי לבצע את הסיווג הסופי או את תחזיות הערכים הרציפים, בהתאם למשימה של הרשת.

Maxpooling2D

השכבה MaxPooling2D היא שכבה ברשתות עצביות קונבולוציוניות (CNN) המשמשת להפחתת מימדים של מפת התכונות (feature map) על ידי בחירת הערך המקסימלי בכל חלון של תאים סמוכים. תפקידה של השכבה הוא להפחית את מספר הפרמטרים והחישובים ברשת, להקטין את הסיכוי לאוברפיטינג ולשמור על התכונות החשובות ביותר של הקלט תוך התעלמות מתכונות פחות משמעותיות. השכבה MaxPooling2D עוזרת לשמור על התכונות הרלוונטיות ומייעלת את תהליך הלמידה של הרשת על ידי הפחתת המורכבות והגברת העמידות לרעשים ושינויים קטנים בקלט.

Batch normalization

השכבה Batch Normalization היא שכבה ברשתות עצביות המשמשת לנרמל את הפלט של שכבה קודמת על ידי התאמת הממוצע והשונות של הנתונים בכל מיני-אצווה (mini-batch). תפקידה של השכבה הוא לייצב ולהאיץ את תהליך האימון על ידי הפחתת התלות בערכים התחלתיים של המשקלים והפחתת בעיות כמו התפוצצות או היעלמות גרדיאנטים. השכבה מבצעת נרמול באמצעות חישוב הממוצע והשונות של כל מיני-אצווה, ולאחר מכן משקלול מחדש של הערכים באמצעות פרמטרים שנלמדים במהלך האימון, מה שמשפר את יכולת הלמידה של הרשת ומאפשר שימוש בפונקציות הפעלה לא רוויות.

UpSampling2D

השכבה UpSampling2D היא שכבה ברשתות עצביות המשמשת להגדלת מימדי מפת התכונות (feature map) על ידי שכפול הנתונים בכל ציר. תפקידה הוא לשחזר את הרזולוציה המקורית של הנתונים לאחר פעולות הפחתת מימדים כמו קונבולוציה או מקס-פולינג. השכבה מבצעת הגדלה על ידי שכפול פיקסלים בשני הצירים, רוחב וגובה, מה שמאפשר לרשת לשמור על מידע מרחבי חשוב ולהגדיל את הרזולוציה של מפת התכונות לצורך פעולות עיבוד נוספות או לשימוש במודלים לייצור תמונות, שיחזור תמונות, או תרגום תמונות.

כעת אסביר מה כל שכבה עושה ספציפית בפרויקט שלי.

1. Conv2D: בשלב הEncoder

שכבת Conv2D לוקחת את התמונה שהעלית ומפעילה עליה מסננים שמזהים תכונות בסיסיות כמו קצוות וקווים. למשל, אם העלית תמונה של חפץ, השכבה הזו תזהה את קווי המתאר והטקסטורות של החפץ. שלב זה קריטי כי הוא מאפשר למודל להפיק מידע ראשוני חיוני מהתמונה שיזוהה בהמשך.

2. BatchNormalization:

לוקחת את הפלט מהשכבה הקודמת ומנרמלת אותו כדי לייצב את האימון. זה אומר שהיא מוודאת שכל התכונות שהופקו נמצאות בטווח ערכים דומה, מה שמונע ערכים קיצוניים. נרמול זה חשוב כי הוא משפר את יציבות האימון והופך את הלמידה ליעילה ומהירה יותר.

3. MaxPooling2D:

מצמצמת את גודל התמונה על ידי שמירת התכונות החשובות ביותר מכל חלון פיקסלים. למשל, אם התמונה שלך כוללת מספר תכונות, השכבה הזו תשמור רק את התכונות המשמעותיות ביותר. הפחתת מימדים בצורה זו מפחיתה את כמות המידע שהמודל צריך לעבד, מה שמייעל את החישובים ומפחית את הסיכון ל-overfitting.

4. UpSampling2D:

שכבת UpSampling2D מגדילה את מימדי התמונה שהוקטנה בשלב ה-Encoder, ומשחזרת אותה לגודל קרוב למקור. זה נעשה על ידי הכפלת הפיקסלים. הגדלת מימדים מאפשרת למודל לשחזר את התמונה המקורית בצורה טובה יותר ומאפשרת למודל להפיק פלט שניתן להציג למשתמש.

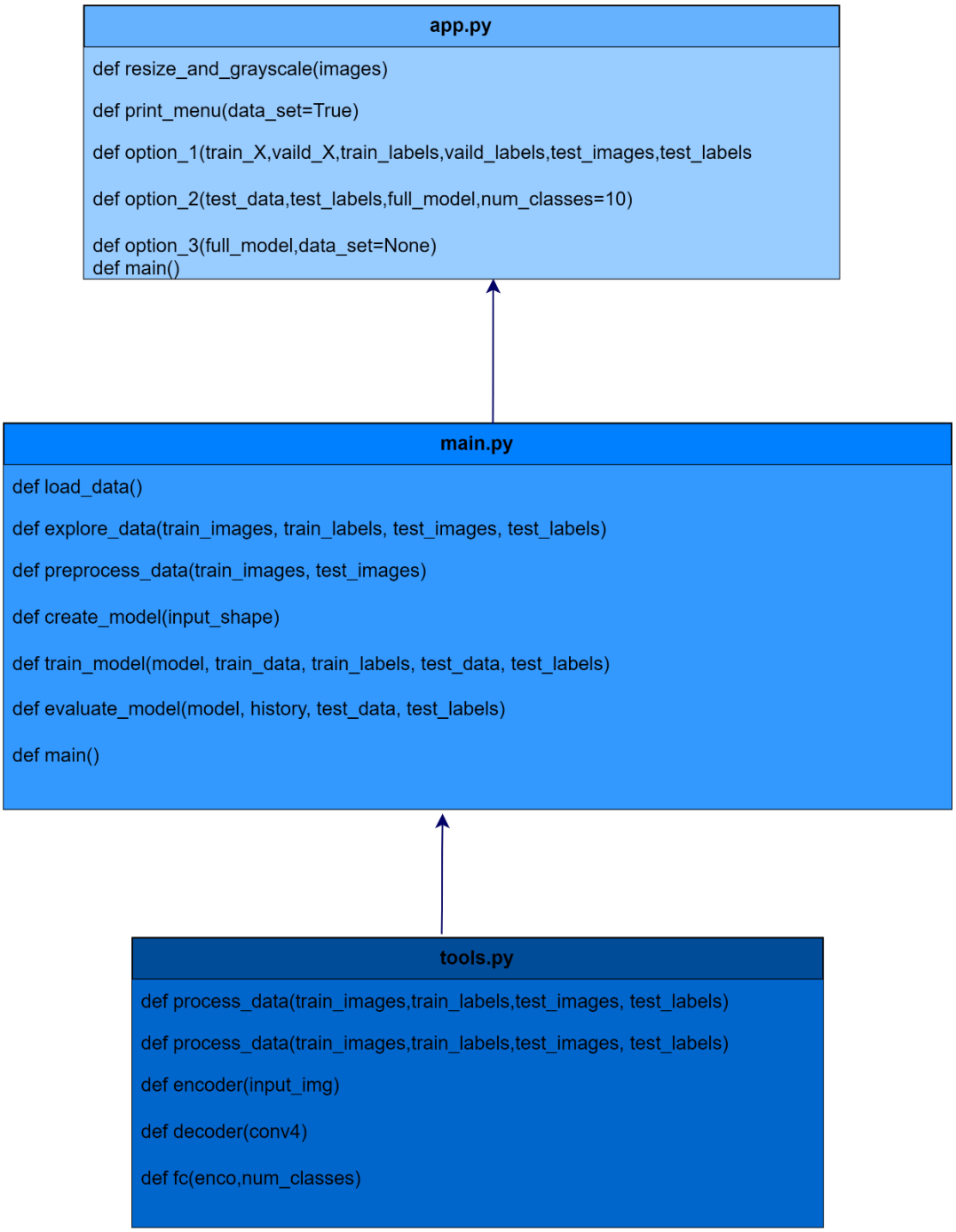
5. Conv2D: בשלב הDecoder

שכבת Conv2D בשלב ה-Decoder מבצעת עיבוד נוסף על התכונות המוגדלות כדי לשחזר את התמונה המקורית בפרטים המדויקים. שלב זה חשוב כי הוא מבטיח שהתמונה המשוחזרת תהיה מדויקת וברורה, מה שמאפשר למודל לבצע תחזיות מדויקות על האובייקטים בתמונה.

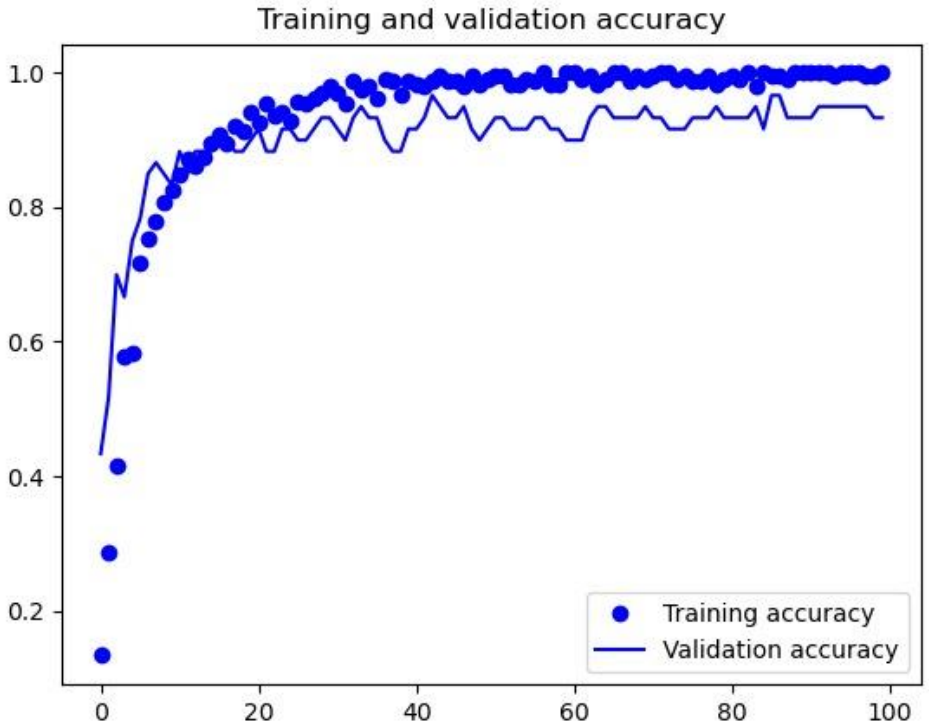
6. Dense (שכבות צפופות):

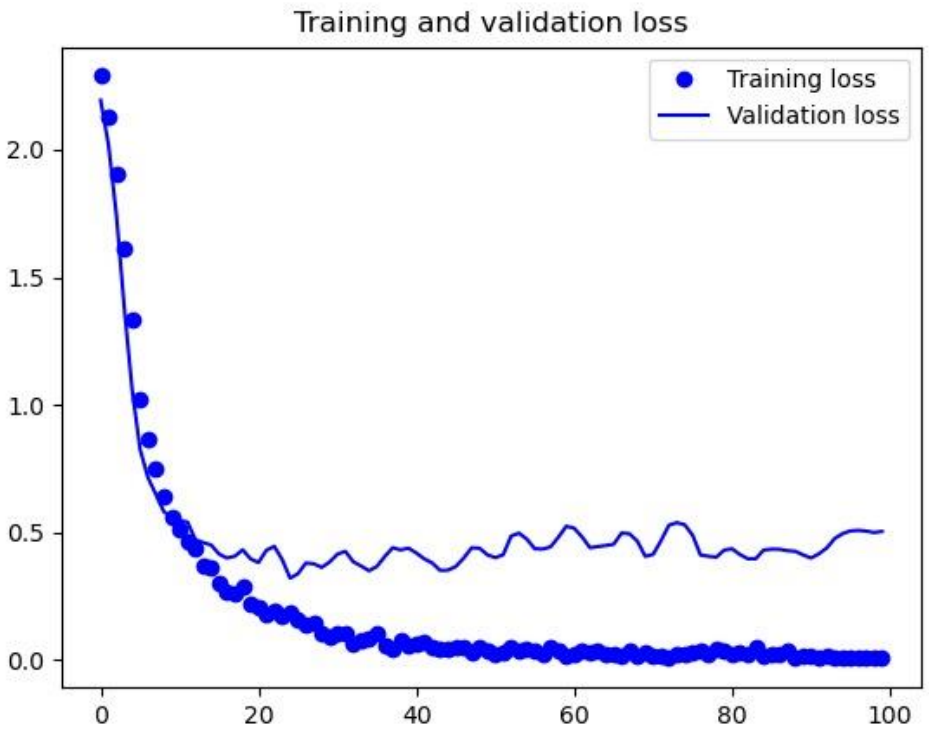
שכבות אלו מבצעות את הסיווג הסופי של האובייקט בתמונה על סמך התכונות שהופקו. הן מחברות את כל המידע שהתמונה מכילה ומחליטות לאיזו קטגוריה היא שייכת. זהו השלב הסופי שבו המודל מחליט על הסיווג של התמונה ומחזיר את התוצאה למשתמש. זה השלב שבו כל העבודה שהשכבות הקודמות עשו מתרכזת לתוצאה אחת.

תיאור UML של המחלקות



דוחות וגרפים המתארים את תוצאות האימון והווידוא מודל :





דוח הכולל ריכוז כל הHyper Parameters:

הHyper Parameters שיש אצלי בפרויקט הם:

Learning Rate, Batch Size, Number of Epochs, Optimizer, Dropout Rate, Activation Function. ארחיב על כל אחד מהם.

learning rate קובע את גודל הצעד שהמודל עושה בכל עדכון משקל במהלך האימון. אם קצב הלמידה גבוה מדי, המודל עשוי להתעלם ממינימום מקומי ולא להתכנס בצורה יציבה. אם קצב הלמידה נמוך מדי, האימון עשוי להיות איטי מאוד ולא להגיע למינימום בתוך זמן סביר. בפרויקט שלי, קביעת קצב הלמידה המתאים היא חיונית להתכנסות מהירה ומדויקת של המודל.

batch size הוא מספר התמונות שעוברות באימון יחד לפני עדכון המשקלים. גודל batch קטן מאפשר עדכונים תכופים יותר של המשקלים, אך עשוי להוביל לתנודתיות באימון. גודל batch גדול מייצב את האימון אך דורש יותר זיכרון ועשוי להאט את התהליך. בפרויקט שלי, בחירת גודל הbatch המתאים משפיעה על יעילות האימון ועל יציבותו.

number of epochs הוא מספר הפעמים שהמודל עובר על כל מערך האימון כולו. מספר epochs קטן מדי עשוי לא לאפשר למודל ללמוד מספיק מהנתונים, בעוד שמספר גדול מדי עשוי להוביל ל-overfitting, כלומר המודל ילמד יתר על המידה את דוגמאות האימון. בפרויקט שלי, מספר הepochs משפיע על היכולת של המודל ללמוד את הדפוסים מהנתונים ולהתכנס לפתרון טוב.

optimizer הוא האלגוריתם שמשמש לעדכון המשקלים במודל במהלך האימון. בחירת האופטימיזטור משפיעה על מהירות ההתכנסות ועל היכולת של המודל למצוא את המשקלים האופטימליים. דוגמאות נפוצות כוללות Adam, SGD ו-RMSprop. בפרויקט שלי, האופטימיזטור שנבחר משפיע על האופן שבו המודל מתכנס ומשיג את הביצועים הטובים ביותר.

dropout rate קובע את אחוז הנוירונים שמבוטלים באופן אקראי בכל שכבה במהלך האימון. Dropout משמש למניעת overfitting על ידי הפחתת התלות של המודל במבנה מסוים של הרשת ומעודד גיוון בפתרונות. בפרויקט שלי, שימוש ב-Dropout עוזר להבטיח שהמודל יוכל להתמודד עם נתונים חדשים ולא ילמד יתר על המידה את נתוני האימון.

activation function קובעת את איך הנוירונים בשכבה מסוימת מעבדים את הפלט שלהם. מדוע זה חשוב: בחירת פונקציית הפעלה כמו ReLU, Sigmoid, או Tanh משפיעה על יכולת המודל ללמוד ולהתכנס. פונקציית ReLU היא הפופולרית ביותר ברשתות נוירונים

עמוקות. בפרויקט שלי, פונקציית ההפעלה משפיעה על איך שהמודל מעבד את הנתונים ומפיק תובנות מהם.

תיעוד כל השינויים שנעשו במודל וב Parameters Hyper לשיפור תוצאות האימון:

מודל סופי:

פרמטרים לאימון	פרמטרים בצורת הרשת ניורנים
Num classes-10	Number of hidden layers: 8
Optimizer Algorithm- Adam	Activation function: Relu
Epoch-100	Dropout=0.5
Batch size-64	Weight initialization- zeros
Learning rate- default- 0.001	

מודל ראשוני- לא סופי:

פרמטרים לאימון	פרמטרים בצורת הרשת ניורנים
Num classes-10	Number of hidden layers: 8
Optimizer Algorithm- Rmsprop	Activation function: Relu
Epoch-150	Dropout=0.2
Batch size-128	Weight initialization- zeros
Learning rate- default- 0.001	

תיעוד והסבר של פונקציית השגיאה:

פונקציית השגיאה (Loss Function) מודדת את ההבדל בין התוצאה הצפויה לתוצאה שחוזה המודל. בפרויקט זה נעשה שימוש בפונקציית Cross-Entropy Loss . אסביר על סוג פונקציה זו ולאחר מכן אקשר לפרויקט שלי.

categorical crossentropy

פונקציית השגיאה categorical crossentropy היא פונקציה נפוצה בלמידת מכונה המשמשת בעיקר למודלים של סיווג מרובה קטגוריות. היא משמשת למדוד את ההבדל בין ההתפלגות הצפויה (הערכים האמיתיים) לבין ההתפלגות החזויה (התחזיות של המודל) של קטגוריות שונות.

1. הכנסת הנתונים וחשיבות הפונקציה:

התפלגות צפויה - עבור כל דוגמה באימון, הנתונים האמיתיים מיוצגים כוקטור one-hot encoded. בוקטור זה, רק הערך המייצג את הקטגוריה הנכונה הוא 1, ושאר הערכים הם 0.

התפלגות חזויה - המודל מייצר עבור כל דוגמה באימון וקטור של הסתברויות, כאשר כל ערך בוקטור מייצג את ההסתברות שהדוגמה שייכת לאותה קטגוריה.

פונקצית השגיאה מודדת בצורה מדויקת את ההבדל בין התחזיות של המודל לערכים האמיתיים, מה שמאפשר להבין כמה טוב המודל מבצע את המשימה שלו.
במהלך האימון, המודל משתמש בתוצאות של פונקציית השגיאה כדי לעדכן את המשקלים שלו. מטרת המודל היא למזער את השגיאה הזו ככל האפשר, מה שמשפר את הדיוק שלו.

2. חישוב השגיאה:

פונקציית השגיאה מחשבת את ההפרש בין ההתפלגות הצפויה להתפלגות החזויה עבור כל קטגוריה ומסכמת את ההפרשים.
הנוסחה של categorical_crossentropy היא:

$$\text{Loss} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i)$$

כאשר Yi הוא הערך הצפוי (1 או 0 ב one-hot encoding) -ו y^i -הוא הערך החזוי (ההסתברות שהמודל נתן לקטגוריה זו).

3. חשיבות פונקצית השגיאה במודל שלי:

בפרויקט שלי, המודל שלי מבצע סיווג של אובייקטים לתוך קטגוריות שונות. פונקציית השגיאה categorical_crossentropy מאפשרת למודל ללמוד ולהתאים את המשקלים שלו כך שהתחזיות שהוא מבצע יהיו קרובות ככל האפשר לערכים האמיתיים. היא מספקת מדד מדויק להערכת הביצועים של המודל בתהליך האימון.

4. הקוד בפרויקט שלי:

```
full_model.compile(loss=keras.losses.categorical_crossentropy,
optimizer=keras.optimizers.Adam(),metrics=['accuracy'])
```

תיעוד והסבר של ייעול ההתכנסות (Optimization) :

אסביר בקצרה מהי פונקציית ייעול ההתכנסות ולאחר מכן ארחיב באיזה אחת השתמשתי בפרויקט שלי.

הסבר על פונקציית יעול ההתכנסות:

פונקציית ייעול ההתכנסות (Optimizer), היא אלגוריתם המשמש לעדכון המשקלים של המודל במהלך האימון במטרה למזער את פונקציית השגיאה. האופטימיזטור משפיע על מהירות ההתכנסות של המודל ועל היכולת שלו להגיע לפתרון אופטימלי.

בפרויקט שלי, אני משתמשת בOptimizer - Adam.

הסבר על האופטימייזר Adam:

Adam (קיצור של Adaptive Moment Estimation) הוא אחד מהאופטימיזטורים הנפוצים והיעילים ביותר בלמידת מכונה. הוא מספק התכנסות מהירה ויציבה, ומתאים את קצב הלמידה לכל פרמטר במודל, מה שמשפר את ביצועי המודל.

Adam משתמש בשני מומנטים שונים:

מומנט ראשון (First Moment Estimation):

המומנט הראשון הוא הגרדיאנט הממוצע.

Adam מחשב את הממוצע המשוקלל של הגרדיאנטים (השינויים המהירים בערכי המשקלים) כדי לשמור על מגמת השיפור של המודל.

מומנט שני (Second Moment Estimation):

המומנט השני הוא ריבוע הגרדיאנט הממוצע.

Adam מחשב את הממוצע המשוקלל של ריבועי הגרדיאנטים כדי להעריך את התפלגות השגיאות הגדולות והקטנות.

הגרדיאנטים מתעדכנים באמצעות שני המומנטים הללו, מה שמאפשר התכנסות מהירה ויציבה גם כאשר פונקציית השגיאה אינה קמורה (non-convex).

בפרויקט שלי, אני משתמשת באופטימיזטור Adam כדי לעדכן את המשקלים של המודל במהלך האימון. הנה שורת הקוד המגדירה את השימוש ב-Adam:

```
full_model.compile(loss=keras.losses.categorical_crossentropy,
optimizer=keras.optimizers.Adam(),metrics=['accuracy'])
```

תיעוד ההתמודדות עם הטיה ושונות (שגיאת אימון ושגיאת מבחן):

הטיה ושונות (Bias and Variance) הם שני מושגים חשובים ב machine learning והם מסבירים את השגיאות במודל. אתחיל בלפרט על כל מושג ולאחר מכן אקשר לפרויקט שלי.

הטיה Bais - עיוות של תוצאות מדידה או הערכה המוביל לפירוש מוטעה שלהן. הטיה גבוהה נגרמת כאשר המודל פשוט מדי ואינו מתאים מספיק את הנתונים. זה מוביל ל - underfitting, כלומר המודל אינו מצליח ללמוד את הדפוסים בנתונים כראוי. המודל מספק תחזיות גרועות גם על נתוני האימון וגם על נתוני הבדיקה.

שונות Variance- שונות מתייחסת לשינויים במודל בעת שימוש בחלקים שונים ממערך נתוני האימון. שונות גבוהה נגרמת כאשר המודל מורכב מדי ומתאים את עצמו בצורה מדויקת מדי לנתוני האימון. זה מוביל ל overfitting, כלומר המודל מצליח על נתוני האימון אך נכשל על נתונים חדשים. המודל מספק תחזיות טובות על נתוני האימון (train) אך תחזיות גרועות על נתוני הבדיקה (test).

כדי להתמודד עם הטיה ושונות משתמשים בטכניקות שונות, אך אני בפרויקט שלי השתמשתי בDropout ו Early Stopping. ארחיב על כל אחת מן הטכניקות ולאחר מכן אסביר מדוע השתמשתי בהן, ואביא הוכחה מן הקוד.

Dropout - במהלך שלב האימון (train), חלק מהנוירונים מבוטלים באופן אקראי בכל שכבה. זה עוזר למנוע overfitting על ידי כך שהמודל לא תלוי במספר גדול של נוירונים כי מספר מסוים של נוירונים, שהמשתמש קובע , לא משפיע.

Early Stopping- עצירת האימון כאשר הביצועים על הנתונים מפסיקים להשתפר. זה מונע overfitting בכך שעצירה מוקדמת מונעת מהמודל להמשיך ללמוד דפוסים ספציפיים של נתוני האימון.

בפרויקט שלי, השימוש בטכניקות של Dropout ו Early Stopping-עוזר להתמודד עם בעיות של הטיה ושונות. טכניקות אלו משפרות את ביצועי המודל ומבטיחות שהוא לא יתאים יתר על המידה לנתוני האימון אלא יהיה מסוגל להתמודד גם עם נתונים חדשים בצורה טובה .

השימוש בDropout בפרויקט שלי מבטיח שהמודל לא יתאים יתר על המידה לנתוני הtrain , על ידי ביטול אקראי של 50% מהנוריונים בכל שכבה במהלך האימון. להלך הקוד

```
model.add(Dropout(0.5))
```

השתמשנו ב EarlyStopping- כדי לעצור את האימון כאשר הביצועים על נתוני האימות מפסיקים להשתפר. בפרמטר patience=10 אנחנו מאפשרים למודל להמשיך להתאמן 10 epoches נוספים אחרי הפסקת השיפור בביצועים לפני ,שעוצרים את האימון.

להלן הקוד

```
early_stopping = EarlyStopping(monitor='val_loss', patience=10,
restore_best_weights=True)
classify_train = full_model.fit(train_X, train_label, batch_size=64,
epochs=100, verbose=1, validation_data=(valid_X, valid_label),
callbacks=[early_stopping])
```

כעת ארחיב על שגיאת אימון ושגיאת מבחן.

שגיאות אימון- שגיאות אימון מתייחסות לאי דיוק של המודל על הdata שהייתה בשימוש בשלב ה train. זוהי המדידה בכמה טוב המודל בללמוד את הדפוס בנתוני האימון.

שגיאות מבחן- שגיאות מבחן מתייחסות לאי-דיוקים של המודל על קבוצת הנתונים שנשמרה לצורך בדיקה (אשר לא שימשה לאימון המודל). מדד זה משמש להערכת היכולת של המודל להכליל את הידע שרכש על נתונים חדשים שלא ראה קודם.

המטרה היא למצוא את האיזון הנכון בין הטיה לשונות כך שהטעות הכללית של המודל תהיה מינימלית על נתוני האימון וגם על נתוני המבחן, ומכאן מגיע הקשר בין הטיה ושונות לבין שגיאות מבחן ואימון.

שגיאות אימון גבוהות ושגיאות מבחן גבוהות :Underfitting -הטיה גבוהה, שונות נמוכה.

שגיאות אימון נמוכות ושגיאות מבחן גבוהות: Overfitting - הטיה נמוכה, שונות גבוהה.

שלב היישום(Software Deployment) :

תיאור והסבר כיצד היישום משתמש במודל:

בפרויקט שלי המודל המיושם נבנה, מאומן ומיושם כדי לסווג ולזהות תמונות. שלבי העבודה עם המודל הם הכנת הנתונים, בניית המודל, קומפילציה של המודל (אופטימיזטור ופונקצית גיאה), אימון המודל, הערכת המודל, שימוש במודל לחיזוי התוצאות ולבסוף תוצר.

המודל נבנה באמצעות שכבות רשת נוירונים, מאומן על נתוני האימון, ומוערך על נתוני האימות/המבחן. המודל משתמש בטכניקות כמו Dropout ו EarlyStopping-כדי למנוע overfitting ולשפר את ביצועיו. לאחר האימון וההערכה, המודל מוכן לבצע תחזיות על נתונים חדשים. כל השלבים הללו משתלבים יחד כדי ליצור יישום למידת מכונה מוצלח שיכול לסווג תמונות בצורה יעילה ומדויקת.

תיאור הטכנולוגיה שעל פיה מומש ממשק המשתמש:

הפרויקט שלי משתמש ב Python-ובספריות רבות כמו Keras, NumPy ו-PIL שהממשק מבוסס על קונסולת הפקודות(Command Line Interface - CLI) .

אסביר כעת על <u>ממשק המשתמש מבוסס קונסולת הפקודות CLI:</u>

ממשק משתמש מבוסס קונסולת הפקודות (CLI - Command Line Interface) הוא סוג של ממשק משתמש המאפשר למשתמשים לקיים אינטראקציה עם התוכנה באמצעות פקודות טקסטואליות המוזנות בקונסולה או בשורת הפקודה.

בCLI המשתמשים יכולים להקליד פקודות בשורת הפקודה כדי לבצע פעולות, התוכנה מציגה תפריטים והנחיות למשתמש כך שיכול לבחור אפשרויות על ידי הקלדת מספר או טקסט מתאים, התוצאות וההודעות מוצגות בטקסט בקונוסלה.

בקוד שלי יש תקשורת זו עם המשתמש:
להלן דוגמא בקוד:

```
def print_menu(data_set=True):
    """
    The function returns the user's choice- between 1 to 4.


    Parameters:
    choice- the choice of the user.
    ---------
    output:
    Returns the user's choice.
    ---------
    """
    if data_set==True:
        data_set = input("Hey, pleast write the dataset you want to
use:")
    print("\nChoose an option")
```

```
    print("1. Train the model")
    print("2. Test the model")
    print("3. Predict an image")
    print("4. Exit")
    choice = input("Enter your choice: ")
    while(choice.isdigit()==False or (int(choice) < 1) or (int(choice)
> 4)):
        print("Sorry, this is not a valid input. ")
        choice = input("Please enter your choice again: ")

    return data_set,int(choice)
```

תיאור קוד הקולט את הנתונים לחיזוי והתאמתם למבנה נתונים המתאים לחיזוי:

הקוד הקולט את הנתונים לחיזוי ומתאים אותם למבנה נתונים המתאים לחיזוי:

```
 predicted_classes = model.predict(test_data)
    predicted_classes = np.argmax(np.round(predicted_classes), axis=1)
```

מדריך למפתח

בפרק זה אסביר על כל קובץ בפרויקט שלי, על הפעולות והמשתנים השונים בו. בפרויקט שלי יש שלושה קבצים שונים כאשר לכל אחד תפקיד שונה ואחריות שונה בפרויקט.

שם הקובץ	תפקיד הקובץ
Main.py	קובץ זה בונה ומפעיל את המודל. הוא מאמן את המודל על הנתונים ומעריך את ביצועי המודל. כמו כן תפקיד ה main הוא גם לבצע את תחזית האובייקט- כלומר מה המודל חוזה כתוצאה.
App.py	קובץ זה מפעיל ומנהל את ממשק המשתמש CLI. הקובץ אחראי על טעינת הנתונים, קריאה לפונקציות עיבוד הנתונים והכנתם- יש לו אחראיות להגדיר את הנתיבים והפונקציות לטיפול בקלטי המשתמש. כמו כן, תפקיד הקובץ הוא גם לנהל את בחירות המשתמש ולהפנות אותו לפונקציות המתאימות.
Tools.py	קובץ זה מכיל פונקציות עזר לעיבוד הנתונים ותחזוקת המודל. פונקציות אלו כולל עיבוד תמונות כמו שינוי גודל וצבע, שמירת נתונים והכנתם לאימון. כמו כן, על הקובץ לעבד את הנתונים בפורמט מתאים למודל.

כעת, ארחיב על תוכן,פעולות ומשתני כל קובץ.

Main.py

קובץ זה נפתח בייבוא הספריות הדרושות לאימון המודל שלי בmachine learning באמצעות Keras ו TensorFlow.

```
import keras
from matplotlib import pyplot as plt
import numpy as np
from tensorflow.keras.datasets import mnist
from keras.models import Model, Sequential
from keras.layers import Input, Dense, Flatten, Dropout, Reshape,
Conv2D, MaxPooling2D, UpSampling2D, Conv2DTranspose, BatchNormalization
from keras.callbacks import ModelCheckpoint
from keras.optimizers import Adadelta, RMSprop, SGD, Adam
from keras import regularizers
from keras import backend as K
```

```
from keras.utils import to_categorical
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
import datetime
```

load data:

```
def load_data():
    (train_images, train_labels), (test_images, test_labels) =
mnist.load_data()
    train_images, train_labels = train_images[:300], train_labels[:300]
    test_images, test_labels = test_images[:300], test_labels[:300]
    return train_images, train_labels, test_images, test_labels
```

הפונקציה הזאת טוענת את הנתונים ממאגרMNIST , שכולל תמונות של ספרות בכתב יד והתיוגים שלהן, ולאחר מכן מבצעת סלקציה של 300 דוגמאות מתוך מערך האימון והבדיקה.

Explore data:

```
def explore_data(train_images, train_labels, test_images, test_labels):
    print("Training set (images) shape:
{shape}".format(shape=train_images.shape))
    print("Test set (images) shape:
{shape}".format(shape=test_images.shape))
    plt.figure(figsize=[5,5])
    plt.subplot(121)
    plt.imshow(np.reshape(train_images[10], (28,28)), cmap='gray')
    plt.title("(Label: " + str(train_labels[10]) + ")")
    plt.figure(figsize=[5,5])
    plt.subplot(121)
    plt.imshow(np.reshape(train_images[11], (28,28)), cmap='gray')
    plt.title("(Label: " + str(train_labels[11]) + ")")
    plt.show()
```

הפונקציה מייצגת ומדפיסה את צורת המערכים של התמונות ממאגר MNIST ומציגה דוגמאות של תמונות מהאימון עם התוויות המתאימות להן, על מנת להציג ולחקור את הנתונים לפני השימוש בהם להדרכת מודל למידת מכונה.

Preprocess data:

```
def preprocess_data(train_images, test_images):
    train_data = train_images.reshape(-1, 28, 28, 1) /
np.max(train_images)
    test_data = test_images.reshape(-1, 28, 28, 1) /
np.max(test_images)
    return train_data, test_data
```

הפונקציה preprocess_data משמשת להכנה ראשונית של נתוני התמונה לצורך השימוש בהם במודל למידת מכונה, על ידי עיבוד התמונות והכנתן לפורמט המתאים לצורך האימון והבדיקה של המודל.

train model:

```
def train_model(model, train_data, train_labels, test_data,
test_labels):
    train_X, valid_X, train_label, valid_label =
train_test_split(train_data, train_labels, test_size=0.2,
random_state=13)
    history = model.fit(train_X, train_label, batch_size=64,
epochs=100, verbose=1, validation_data=(valid_X, valid_label))
    return history
```

הפונקציה train_model מקבלת מודל למידת מכונה, מערכים של נתוני אימון ובדיקה, ומבצעת אימון של המודל על נתוני האימון עם בדיקה על נתוני האימות. היא מחזירה את ההיסטוריה של האימון, שמכילה מידע על ביצועי המודל במהלך האימון.

evaluate model:

```
def evaluate_model(model, history, test_data, test_labels):
    accuracy = history.history['accuracy']
    val_accuracy = history.history['val_accuracy']
    loss = history.history['loss']
    val_loss = history.history['val_loss']
    epochs = range(len(accuracy))

    plt.plot(epochs, accuracy, 'bo', label='Training accuracy')
    plt.plot(epochs, val_accuracy, 'b', label='Validation accuracy')
    plt.title('Training and validation accuracy')
    plt.legend()
    plt.figure()
    plt.plot(epochs, loss, 'bo', label='Training loss')
    plt.plot(epochs, val_loss, 'b', label='Validation loss')
    plt.title('Training and validation loss')
    plt.legend()
    plt.show()

    test_eval = model.evaluate(test_data, test_labels, verbose=0)
    print('Test loss:', test_eval[0])
    print('Test accuracy:', test_eval[1])

    predicted_classes = model.predict(test_data)
    predicted_classes = np.argmax(np.round(predicted_classes), axis=1)
```

```
    correct = np.where(predicted_classes == test_labels)[0]
    plt.figure(figsize=(10,10))
    for i, correct in enumerate(correct[:9]):
        plt.subplot(3, 3, i+1)
        plt.imshow(test_data[correct].reshape(28, 28), cmap='gray',
interpolation='none')
        plt.title(f"Predicted {predicted_classes[correct]}, Class
{test_labels[correct]}")
        plt.tight_layout()
    plt.show()

    target_names = [f"Class {i}" for i in range(10)]
    print(classification_report(test_labels, predicted_classes,
target_names=target_names))
```

הפונקציה `evaluate_model` מבצעת את הביצועים הבאים:

הצגת גרפים של דיוק ואובדן באימון ובאימון האמינות (validation) של המודל.

מדידת האובדן והדיוק של המודל על נתוני הבדיקה.

הצגת תמונות מהנתונים שבהם המודל טעה בזיהוי המחלקה.

הצגת דו"ח מפורט של תוצאות הביצועים של המודל, כולל דיוק, רגישות ו-F1 סקור (F1 סקור (F1 score) הוא מדד המשלב בין רגישות (recall) ובין דיוק (precision) של מודל למידת מכונה. המדד נוצר כדי לספק מדידה יחידה של ביצועי המודל בזיהוי וסיווג נכונים של מחלקות בנתונים הבדיקה.) עבור כל מחלקה בנתוני הבדיקה.

<u>Main:</u>

```
    train_images, train_labels, test_images, test_labels = load_data()
    explore_data(train_images, train_labels, test_images, test_labels)
    train_data, test_data = preprocess_data(train_images, test_images)
    model = create_model((28, 28, 1))
    history = train_model(model, train_data, train_labels, test_data,
test_labels)
    evaluate_model(model, history, test_data, test_labels)

if __name__ == "__main__":
    main()
```

הפונקציה `main()` מקשרת ומפעילה את כל תהליך האימון והערכה של מודל למידת מכונה במסגרת הבאה:

1. **טעינת הנתונים**: היא טוענת את נתוני האימון והבדיקה ממקור הנתונים המתאים, כמו קבצי תמונות ותוויות או מבסיס נתונים.

2. **גילוי הנתונים**: פונקציה זו מבצעת חקירה ראשונית של הנתונים הטעונים, על מנת להבין את מאפייניהם העיקריים ולוודא שהם תקינים לשימוש.

3. **עיבוד הנתונים**: מטפלת בעיבוד ובטיפול הנתונים, כולל התאמת גודל התמונה והמרה למבנה המתאים לטובת תהליך האימון של המודל.

4. **יצירת המודל**: מייצרת את מודל הלמידה המכונה עם הארכיטקטורה והפרמטרים הנדרשים, כולל קביעת שכבות נוירונים, פונקציות הפעולה, וכו'.

5. **אימון המודל**: מאמנת את המודל על נתוני האימון, כך שהוא יכול ללמוד את הקשרים ואת התפקודים בין הנתונים, ובודקת את ביצועיו.

6. **הערכה והדפסת התוצאות**: משווה את ביצועי המודל על נתוני הבדיקה ומדפיסה את התוצאות, כולל דיוק, רגישות ו-F1 סקור לכל מחלקה, ומציגה פרטים נוספים כמו גרפים של ביצועי האימון והאובדן.

App.py

קובץ זה נפתח בייבוא הספריות הדרושות לאימון המודל שלי בmachine learning.

```
import keras
from matplotlib import pyplot as plt
import numpy as np
import gzip
# %matplotlib inline
from keras.models import Model
from keras.optimizers import RMSprop
from keras.layers import
Input,Dense,Flatten,Dropout,Reshape,Conv2D,MaxPooling2D,UpSampling2D,Co
nv2DTranspose
from keras.layers import BatchNormalization
from keras.models import Model,Sequential
from keras.callbacks import ModelCheckpoint
from keras.optimizers import Adadelta, RMSprop,SGD,Adam
from keras import regularizers
from keras import backend as K
from keras.utils import to_categorical
from sklearn.model_selection import train_test_split
from tools import *
from tensorflow.keras.datasets import mnist
import datetime
from tools import *
```

```
from sklearn.metrics import classification_report
import os
from PIL import Image
from tensorflow.keras.datasets import cifar10

import numpy as np
from PIL import Image
```

resize and grayscale

```
def resize_and_grayscale(images):
    # Initialize an array for the processed images
    processed_images = np.zeros((images.shape[0], 28, 28, 1),
dtype=np.float32)

    for i in range(images.shape[0]):
        # Convert each image to PIL Image, resize it and convert to
grayscale
        img = Image.fromarray(images[i])
        img =img.resize((28, 28), Image.LANCZOS).convert('L')

        # Convert back to an array and normalize it
        img = np.array(img, dtype=np.float32) / 255.0

        # Add an extra dimension for channel
        processed_images[i, :, :, 0] = img

    return processed_images
```

הפונקציה מעבדת כל תמונה במערך התמונות שהפונקציה מקבלת, משנה את גודלה ל28 על 28 פיקסלים, ממירה אותה לגווני אפור, מנרמלת את הערכים שלה לטווח של 0 ל-1 ומוסיפה מימד נוסף. התוצאה היא מערך של תמונות מעובדות המוכנות לשימוש במודל.

Print menu

```
def print_menu(data_set=True):
    """
    The function returns the user's choice- between 1 to 4.


    Parameters:
    choice- the choice of the user.
    ---------
    output:
    Returns the user's choice.
```

```
    ---------
    """
    if data_set==True:
        data_set = input("Hey, please write the dataset you want to
use:")
    print("\nChoose an option")
    print("1. Train the model")
    print("2. Test the model")
    print("3. Predict an image")
    print("4. Exit")
    choice = input("Enter your choice: ")
    while(choice.isdigit()==False or (int(choice) < 1) or (int(choice)
> 4)):
        print("Sorry, this is not a valid input. ")
        choice = input("Please enter your choice again: ")

    return data_set,int(choice)
```

הפונקציה מציגה למשתמש תפריט של ארבע אפשרויות ומבקשת ממנו לבחור באחת מהן. אם נדרש, היא גם מבקשת ממנו לציין את ערכת הנתונים שהוא רוצה להשתמש בה. היא מוודאת שהבחירה תקינה (בין 1 ל-4) ומחזירה את ערכת הנתונים והבחירה של המשתמש.

<u>option_1/2/3</u>

בקוד שלי מופיעות שלוש פונקציות המתאימות לשלושת האפשרויות מהן יכול לבחור המשתמש – אופציה 1 ל1 וכו.

כל אחת מהפונקציות עובדת על המטרה שלו- אופציה 1 על אימון המודל, אופציה 2 על בדיקת המודל ואופציה 3 על חיזוי המודל.

```
def
option_1(train_X,valid_X,train_labels,valid_labels,test_images,test_lab
els):
    model_name='base_model'
    batch_size = 128
    epochs = 2
    inChannel = 1
    x, y = 28, 28
    input_img = Input(shape = (x, y, inChannel))
    num_classes = 10



    #Model_1 (Encoder-Decoder)- purpose- get the best parameter for the
encoder layer


    autoencoder = Model(input_img, decoder(encoder(input_img)))
```

```
    autoencoder.compile(loss='mean_squared_error', optimizer =
RMSprop())
    autoencoder_train = autoencoder.fit(train_X, train_X,
batch_size=batch_size,epochs=epochs,verbose=1,validation_data=(valid_X,
valid_X))

    autoencoder.save_weights('autoencoder.weights.h5')

    #model_2 (Encoder-Fully_connected) purpose - use the best parameter
in the encoder to predcit the output
    encode = encoder(input_img)
    full_model = Model(input_img,fc(encode,num_classes))

    # Model_2 train only fc layer
    # define the saved parameters in the encoder layers
    for l1,l2 in zip(full_model.layers[:19],autoencoder.layers[0:19]):
        l1.set_weights(l2.get_weights())

    #freeze the weights on the encoder layer
    for layer in full_model.layers[0:19]:
        layer.trainable = False

    full_model.compile(loss=keras.losses.categorical_crossentropy,
optimizer=keras.optimizers.Adam(),metrics=['accuracy'])
    classify_train = full_model.fit(train_X, train_labels,
batch_size=64,epochs=2,verbose=1,validation_data=(valid_X,
valid_labels))


    #Model_2 train both encoder and fc layers
    #unfreeze the encoder layer
    for layer in full_model.layers[0:19]:
        layer.trainable = True

    full_model.compile(loss=keras.losses.categorical_crossentropy,
optimizer=keras.optimizers.Adam(),metrics=['accuracy'])

    classify_train = full_model.fit(train_X, train_labels,
batch_size=64,epochs=2,verbose=1,validation_data=(valid_X,
valid_labels))
    full_model.save_weights('classification_complete.weights.h5')
    print("Done! \nThe model is successfully trained.")
    return full_model

def option_2(test_data,test_labels, full_model,num_classes=10):

    predicted_classes = full_model.predict(test_data)
    predicted_classes = np.argmax(np.round(predicted_classes),axis=1)
    test_labels_ordered=np.argmax(np.round(test_labels),axis=1)
```

```
    target_names = ["Class {}".format(i) for i in range(num_classes)]
    print(classification_report(test_labels_ordered, predicted_classes,
target_names=target_names))


def option_3(full_model,data_set=None):
    folder_path="predicted_images"
    image_size=(28, 28)
    #chatgpt output- should edit
    images = os.listdir(folder_path)

    for image_name in images:
        # Ensure only images are processed
        if image_name.lower().endswith(('.png', '.jpg', '.jpeg')):
            # Load the image
            img_path = os.path.join(folder_path, image_name)
            img = Image.open(img_path).convert('L')

            # Resize and preprocess the image
            img = img.resize(image_size)
            img_array = np.array(img) / 255.0  # Scale pixel values to
[0, 1]
            if data_set=="cifar10":
                img_array = np.expand_dims(img_array, axis=-1)
            img_array = np.expand_dims(img_array, axis=0)  # Model
expects batch dimension

            # Predict the class
            predictions = full_model.predict(img_array)
            predicted_class = np.argmax(predictions, axis=1)


            plt.imshow(img)
            if data_set=="cifar10":
                cifar10_classes = {
                    0: 'airplane',
                    1: 'car',
                    2: 'bird',
                    3: 'cat',
                    4: 'deer',
                    5: 'dog',
                    6: 'frog',
                    7: 'horse',
                    8: 'ship',
                    9: 'truck'
                }
                plt.title(f"Predicted class:
{cifar10_classes[predicted_class[0]]}")
```

```
            else:
                plt.title(f"Predicted class: {predicted_class[0]}")
            plt.axis('off')  # Turn off axis numbers and ticks
            plt.show()
    return
```

main

```
def main():
    data_set,user_choice = print_menu()

    # model= nural_network.define_model()
    # trainX, trainY, validationX,
validationY=changing_data.send_train()
    # testX, testY=changing_data.send_test()
    if data_set=="mnist":
        (train_images, train_labels), (test_images, test_labels) =
mnist.load_data()
    if data_set=="cifar10":
        (train_images, train_labels), (test_images, test_labels) =
cifar10.load_data()
        train_images = resize_and_grayscale(train_images)
        test_images = resize_and_grayscale(test_images)


    train_X,valid_X,train_label,valid_label,test_images,test_labels=pro
cess_data(train_images, train_labels,test_images, test_labels)


    while not user_choice== 4:
        if user_choice == 1:
            full_model=option_1(train_X,valid_X,train_label,valid_label
,test_images,test_labels)
        elif user_choice == 2:
            option_2(test_images, test_labels,full_model)
        elif user_choice == 3:
            option_3(full_model,data_set)
        data_set,user_choice  = print_menu(data_set=data_set)


if __name__ == "__main__":
    main()
```

הקוד מציג תפריט למשתמש, טוען את נתוני האימון והבדיקה, מעבד את הנתונים, ומבצע פעולות שונות על פי בחירת המשתמש, כולל אימון המודל, בדיקתו וחיזוי תמונות.

Tools.py

קובץ זה נפתח בייבוא הספריות הדרושות לאימון המודל שלי בmachine learning.

```
import keras
from matplotlib import pyplot as plt
import numpy as np
import gzip
# %matplotlib inline
from keras.models import Model
from keras.optimizers import RMSprop
from keras.layers import
Input,Dense,Flatten,Dropout,Reshape,Conv2D,MaxPooling2D,UpSampling2D,Co
nv2DTranspose
from keras.layers import BatchNormalization
from keras.models import Model,Sequential
from keras.callbacks import ModelCheckpoint
from keras.optimizers import Adadelta, RMSprop,SGD,Adam
from keras import regularizers
from keras import backend as K
from keras.utils import to_categorical
from sklearn.model_selection import train_test_split
from tools import *
from tensorflow.keras.datasets import mnist
import datetime
```

<u>process data</u>

```
def process_data(train_images,train_labels,test_images, test_labels):


    #4. preprocessing data
    train_data = train_images.reshape(-1, 28,28, 1)
    test_data = test_images.reshape(-1, 28,28, 1)
    train_data.shape, test_data.shape

    # 4.1 Nomralization
    train_data = train_data / np.max(train_data)
    test_data = test_data / np.max(test_data)
```

```
    train_Y_one_hot = to_categorical(train_labels)
    test_Y_one_hot = to_categorical(test_labels)
    # Display the change for category label using one-hot encoding
    print('Original label:', train_labels[0])
    print('After conversion to one-hot:', train_Y_one_hot[0])

    train_X,valid_X,train_label,valid_label =
train_test_split(train_data,train_Y_one_hot,test_size=0.2,random_state=
13)

    return
train_X,valid_X,train_label,valid_label,test_images,test_labels
```

הפונקציה משנה את צורת הנתונים, מנרמלת אותם, ממירה את התוויות לפורמט של

One hot encoding

מפצלת את נתוני האימון לסט של אימון וולידציה

Train and validation

ומחזירה את הנתונים המעובדים והמוכנים לאימון המודל ולבדיקתו.

הפונקציה מדפיסה את התווית המקורית של הדוגמה הראשונה בנתוני האימון ואת הייצוג שלה לאחר ההמרה ל

.one-hot encoding

הסבר על משתנים חשובים

בפרויקט שלי המשתנים המרכזיים כוללים את נתוני Test and train ,,, תוויות ה One hot encoding

פרמטרים לאימון המודלים, מודלים והגדרותיהם, אובייקטים של היסטוריות אימון Test תחזיות ובדיקת ביצועים. אפרט על מספר משתנים כעת.

<u>נתוני אימון ובדיקה</u>

-Train_labels, train_images

שני משתנים המכילים את תמונות ותוויות נתוני האימון.

-test_images, test_labels
שני משתנים המכילים תמונות ותוויות נתוני הבדיקה

-train_X, valid_X, train_label, valid_label

נתוני האימון והוולידציה לאחר עיבוד ופיצול

<u>עיבוד נתונים מקדים</u>

-train_Y_one_hot, test_Y_one_hot

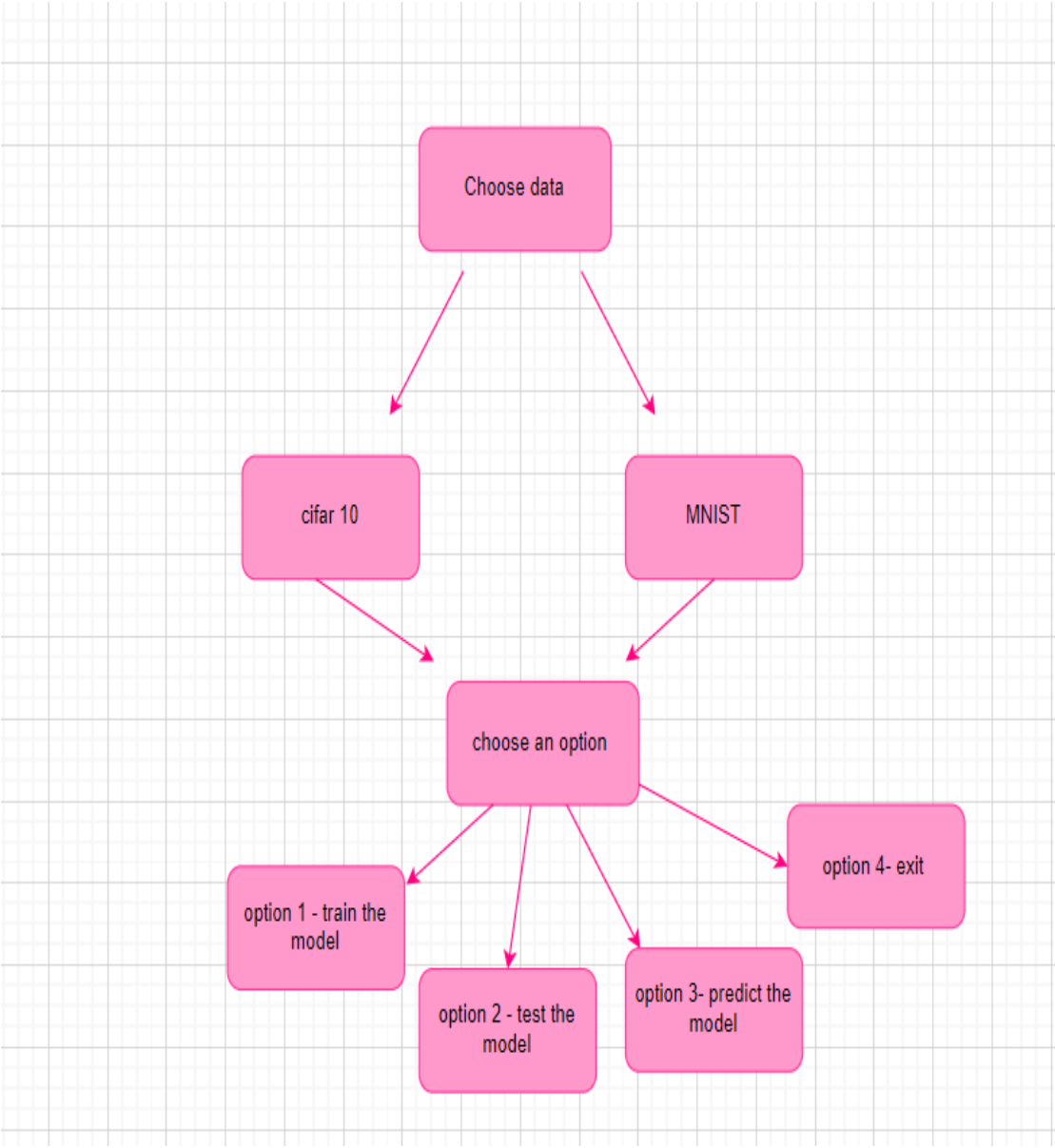
תוויות האימון והבדיקה בפורמט one-hot encoding

<u>פרמטרים לאימון המודלים</u>

גודל הבאטץ' (קבוצת הדוגמאות) לאימון המודל -batch_size
מספר המחזורים של אימון המודל -Epochs
פרמטרים הקשורים לגודל וצורת התמונות -inChannel, x, y
מספר המחלקות (קטגוריות) במודל -num_classes

המדריך למשתמש

תרשים מסכים המתאר את היררכיית המסכים והמעברים ביניהם:



תפקיד כל מסך:

Choose data- מסך הפתיחה!

```
Hey, pleast write the dataset you want to use:cifar10
```

המסך הוא חלק מממשק המשתמש מבוסס קונסולת הפקודות CLI בפרויקט.

מסך זה נועד לאפשר למשתמש לבחור את מערכת הנתונים שבה הוא רוצה להשתמש.

המשתמש יקליד את שם מערכת הנתונים, למשל cifar10 או mnist.

מסך זה הוא נקודת הניווט, רק בעזרתו יכול המשתמש להמשיך הלאה לשאר הממשק ולקבלת התוצאות שלו. הוא נקודת ההתחלה של המשתמש

Choose an option

המסך הוא חלק מממשק המשתמש מבוסס קונסולת הפקודות CLI בפרויקט.

מסך זה מופיע לאחר המסך של Choose data.

המסך הזה מציג למשתמש תפריט של אופציות לבחירה, וכל בחירה מפעילה פונקציה אחרת ביישום. המסך הזה מאפשר למשתמש להפעיל פונקציות שונות ביישום כגון אימון המודל, בדיקת המודל, ביצוע תחזיות, או יציאה מהיישום. תפקידו של המסך הוא לנהל את זרימת התוכנית בהתאם לבחירות המשתמש ולהבטיח שהקלט שהוזן תקין.

התפריט כולל ארבע אפשרויות לבחירה -

Choose an option

1. Train the model

2. Test the model

3. Predict an image

4. Exit

המשתמש מתבקש להזין את הבחירה שלו באמצעות מספר המתאים לאפשרות הרצויה -

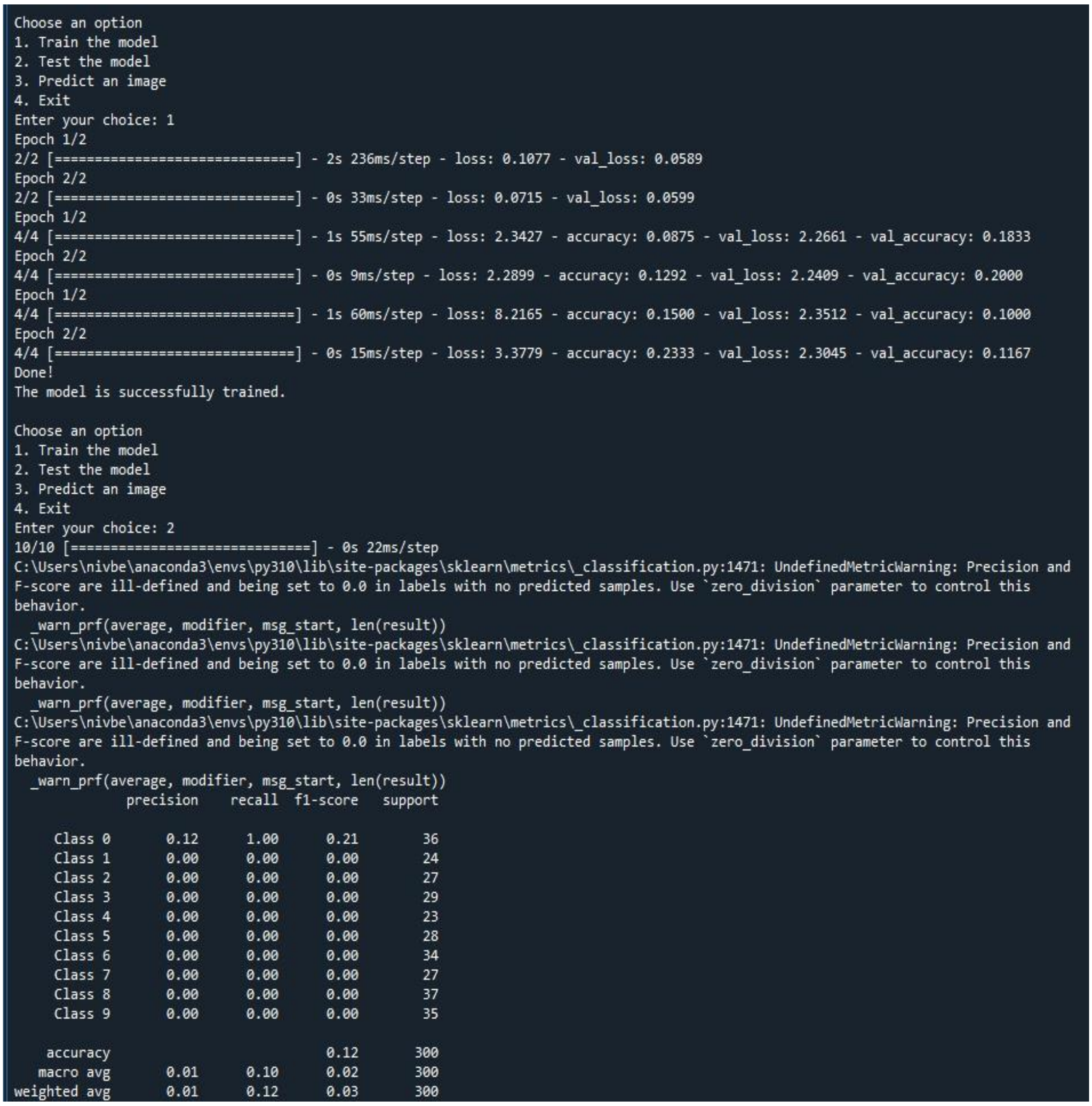
אם המשתמש יזין ערך שאינו מספר או מספר מחוץ לטווח האפשרויות (1-4), תוצג - הודעת שגיאה והמשתמש יתבקש להזין שוב את הבחירה שלו.

אם המשתמש בוחר באפשרות `1`, התוכנית מפעילה פונקציה שמאמנת את המודל על - הנתונים שנבחרו.

אם המשתמש בוחר באפשרות `2`, התוכנית מפעילה פונקציה שבודקת את ביצועי - המודל על סט הבדיקה.

אם המשתמש בוחר באפשרות `3`, התוכנית מפעילה פונקציה שמבצעת תחזית על - תמונה חדשה שהמשתמש מספק.

אם המשתמש בוחר באפשרות `4`, התוכנית יוצאת מהיישום.

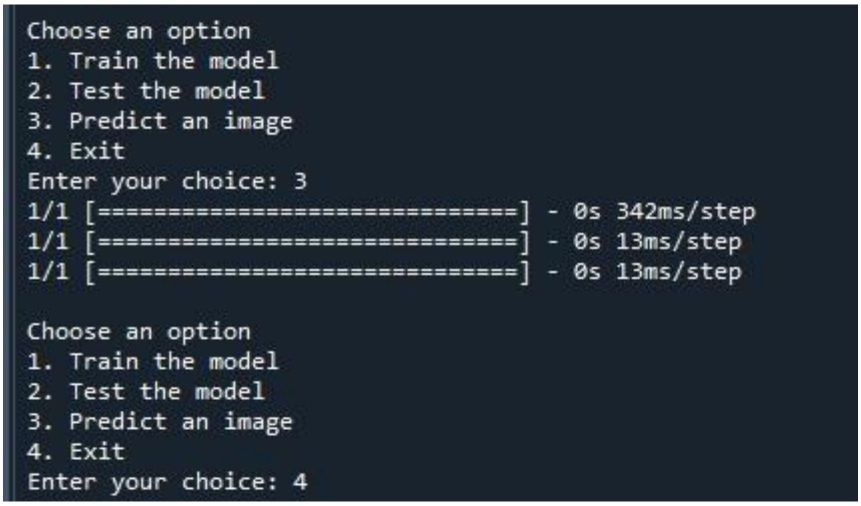


```
Choose an option
1. Train the model
2. Test the model
3. Predict an image
4. Exit
Enter your choice: 1
Epoch 1/2
2/2 [==============================] - 2s 236ms/step - loss: 0.1077 - val_loss: 0.0589
Epoch 2/2
2/2 [==============================] - 0s 33ms/step - loss: 0.0715 - val_loss: 0.0599
Epoch 1/2
4/4 [==============================] - 1s 55ms/step - loss: 2.3427 - accuracy: 0.0875 - val_loss: 2.2661 - val_accuracy: 0.1833
Epoch 2/2
4/4 [==============================] - 0s 9ms/step - loss: 2.2899 - accuracy: 0.1292 - val_loss: 2.2409 - val_accuracy: 0.2000
Epoch 1/2
4/4 [==============================] - 1s 60ms/step - loss: 8.2165 - accuracy: 0.1500 - val_loss: 2.3512 - val_accuracy: 0.1000
Epoch 2/2
4/4 [==============================] - 0s 15ms/step - loss: 3.3779 - accuracy: 0.2333 - val_loss: 2.3045 - val_accuracy: 0.1167
Done!
The model is successfully trained.

Choose an option
1. Train the model
2. Test the model
3. Predict an image
4. Exit
Enter your choice: 2
10/10 [==============================] - 0s 22ms/step
C:\Users\nivbe\anaconda3\envs\py310\lib\site-packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\nivbe\anaconda3\envs\py310\lib\site-packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\nivbe\anaconda3\envs\py310\lib\site-packages\sklearn\metrics\_classification.py:1471: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

     Class 0       0.12      1.00      0.21        36
     Class 1       0.00      0.00      0.00        24
     Class 2       0.00      0.00      0.00        27
     Class 3       0.00      0.00      0.00        29
     Class 4       0.00      0.00      0.00        23
     Class 5       0.00      0.00      0.00        28
     Class 6       0.00      0.00      0.00        34
     Class 7       0.00      0.00      0.00        27
     Class 8       0.00      0.00      0.00        37
     Class 9       0.00      0.00      0.00        35

    accuracy                           0.12       300
   macro avg       0.01      0.10      0.02       300
weighted avg       0.01      0.12      0.03       300
```



```
Choose an option
1. Train the model
2. Test the model
3. Predict an image
4. Exit
Enter your choice: 3
1/1 [==============================] - 0s 342ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 13ms/step

Choose an option
1. Train the model
2. Test the model
3. Predict an image
4. Exit
Enter your choice: 4
```

דרישות ההתקנה בסביבת העבודה, הוראות התקנה ואילו קבצים נדרשים, באילו תיקיות וכדומה

בפרויקט שלי אני כתבתי בסביבת העבודה visual studio code,
כך שעל המשתמש להתקין סביבה זו במחשב שלו על מנת להריץ את הפרויקט.

https://code.visualstudio.com/download

בקישור זה ניתן להוריד את סביבת העבודה בקלות. לחץ על הקישור ובחר את המערכת בה אתה משתמש- ולחץ על הורדה למערכת זו.

על מנת להריץ את הפרויקט שלי, למשתמש צריכות להיות כל הספריות בהן השתמשתי מותקנות בסביבת העבודה שלו על מנת שיוכל להריץ את הפרויקט ושהפרויקט יעבוד עבורו. אסביר בקצרה על כל ספריה ואיך להוריד אותה.

keras

```
import keras
```

הגדרת הספריה- ספרייה ללמידת מכונה ויצירת מודלים של רשתות נוירונים.

התקנת הספריה : כתיבת הקוד - pip install keras

matplotlib.pyplot

```
from matplotlib import pyplot as plt
```

הגדרת הספריה- ספרייה ליצירת גרפים ב Python.

התקנת הספריה : כתיבת הקוד - pip install matplotlib

numpy

```
import numpy as np
```

הגדרת הספריה- ספרייה לחישובים מתמטיים ועבודה עם מערכים מרובי ממדים.

התקנת הספריה : כתיבת הקוד - pip install numpy

gzip

```
import gzip
```

הגדרת הספריה- ספרייה מובנית בפייתון לדחיסת ופריסת קבצים בפורמט GZIP.

התקנה- אין צורך בהתקנה כי הספריה מובנית בפייתון.

Sklearn(scikit- learn)

```
from sklearn.model_selection import train_test_split
```

הגדרת הספריה- ספרייה ללמידת מכונה, הכוללת כלים לסיווג, רגרסיה ו clustering.

התקנת הספריה : כתיבת הקוד - pip install scikit-learn

tools

```
from tools import
```

הגדרת הספריה - ספריה פנימית לפונקציות עזר.
התקנה - אין צורך בהתקנה משום שמדובר בקבצים פנימיים בפרויקט.

tensorflow.keras.datasets

```
from tensorflow.keras.datasets import mnist
from tensorflow.keras.datasets import cifar10
```

הגדרת הספריה – מודל של TensorFlow המכיל מערכי נתונים.

התקנת הספריה: כתיבת קוד- pip install tensorflow

datetime

```
import datetime
```

הגדרת הספריה- ספריי מובנית בפייתון לעבודה עם תאריכים ושעות.

התקנה- אין צורך בהתקנה כי הספריה מובנית בפייתון.

PIL(Pillow)

```
from PIL import Image
```

הגדרת הספריה – ספרייה לעיבוד תמונה.

התקנת הספריה: כתיבת קוד-pip install pillow

os

```
import os
```

הגדרת הספריה- ספריה מובנית בפייתון למערכת הפעלה וממשק עם קבצים ומערכת הקבצים.

התקנה- אין צורך בהתקנה כי הספריה מובנית בפייתון.

רפלקציה

הפרויקט עבורי היה חוויה מעשירה ומלמדת, שבה רכשתי ידע טכני חשוב והתמודדתי עם אתגרים מורכבים.

העבודה על הפרויקט חוויה מרתקת ומלמדת. השתמשתי בטכנולוגיות שונות בלמידת מכונות ועיבוד נתונים והצלחתי לבנות מערכת שתבצע את המטרה שהצבתי לעצמי. העבודה כללה תהליכים מורכבים ושונים של בניית מודלים, עיבוד נתונים ותקשורת עם המשתמש.

קיבלתי מהפרויקט ידע טכני רב על ספריות שונות ועל כיצד מעבדים תמונות שונות וידע זה ישמש אותי גם בפרויקטים עתידיים.

אני רוכשת מן הפרויקט כלים רבים שילכו איתי לחיים כמו היכולת לנתח ולפתור בעיות מורכבות בלמידת מכונה, כולל התמודדות עם הצורך לשפר את המודל ועם בעיות שונות בקוד. למדתי לא לוותר לעצמי גם כשקשה ולהיעזר בחברים ובמורה שלי, לדעת לבקש ולקבל עזרה מאחרים ולשים את האגו בצד. התנסיתי בניהול פרויקט מקצה לקצה, מתכנון ועד יישום ובדיקה ובכלי זה אוכל להשתמש בחיי העתידיים.

במהלך הפרויקט שלי התמודדתי עם קשיים ואתגרים שונים. הייתי צריכה לחשוב על רעיון לפרויקט לבצע אותו עם ידע כמעט מינימלי בנושא. הייתי צריכה ללמוד את הנושא ולפתח את הפרויקט אותו אני רוצה לעשות- לחשוב על דרך ביצוע ולפתח מודלים שונים. גם לאחר שחשבתי והגיתי רעיונות התמודדתי עם קשיים בקוד- התקנת כל הספריות שהיו חסרות היה האתגר הראשוני. היה מאתגר למצוא את האיזון הנכון בין התאמת המודל לנתוני האימון לבין שמירה על היכולת להכליל לנתונים חדשים. כמו כן, גם עיבוד התמונות והתאמתן למודל היוו אתגר טכני. במהלך כל שלבי עבודתי, הייתי צריכה להתמודד עם שגיאות בקוד ולדעת לטפל בהן.

המסקנות שלי מהפרויקט היו שלמידת מכונה דורשת ממני כתלמידה הבנה מעבר לרק שפת תכנות. למדתי שניסוי וטעייה הם חלק בלתי נפרד מהתהליך, שתמיד צריך להמשיך לשפר ולדייק את המודל שלי- שחלק מהלמידה היא בטעויות שעשיתי בדרך.

אילו הייתי מתחילה היום הייתי משקיעה יותר זמן בתכנון ראשוני של הפרויקט לפני ההתחלה, הייתי משנה את התקשורת עם המשתמש לצורה טיפה יותר מתקדמת. באופן כללי, מרגיש לי שהייתה צריכה לפרוס את זמן העבודה ליותר זמן, דבר שהיה מוריד ממני לחץ.

אם יכולתי לעבוד בצוות ובשיתוף פעולה עם אנשים נוספים בעלי ידע בתחום הזה, יכולתי בעזרתם להגיע לתובנות חדשות ולשיפור המודל.

לסיכום, אני שמחה שיצא לי לחוות עבודה בתחום חדשני ומתקדם שכזה, דבר שלא רבים חווים בגיל שלי, אני מרגישה שלמדתי המון גם בפן לימודי וגם בפן אישי.

ביבליוגרפיה

1. Tensorflow- explanation on keras

https://www.tensorflow.org/guide/keras

2. January 2024, Jamell Alvah Samuels explanation on one hot encoding

https://www.researchgate.net/publication/377159812_One-Hot_Encoding_and_Two-Hot_Encoding_An_Introduction

3. Diagrams- where I did UML

https://app.diagrams.net/