



PRISM

for your monochrome

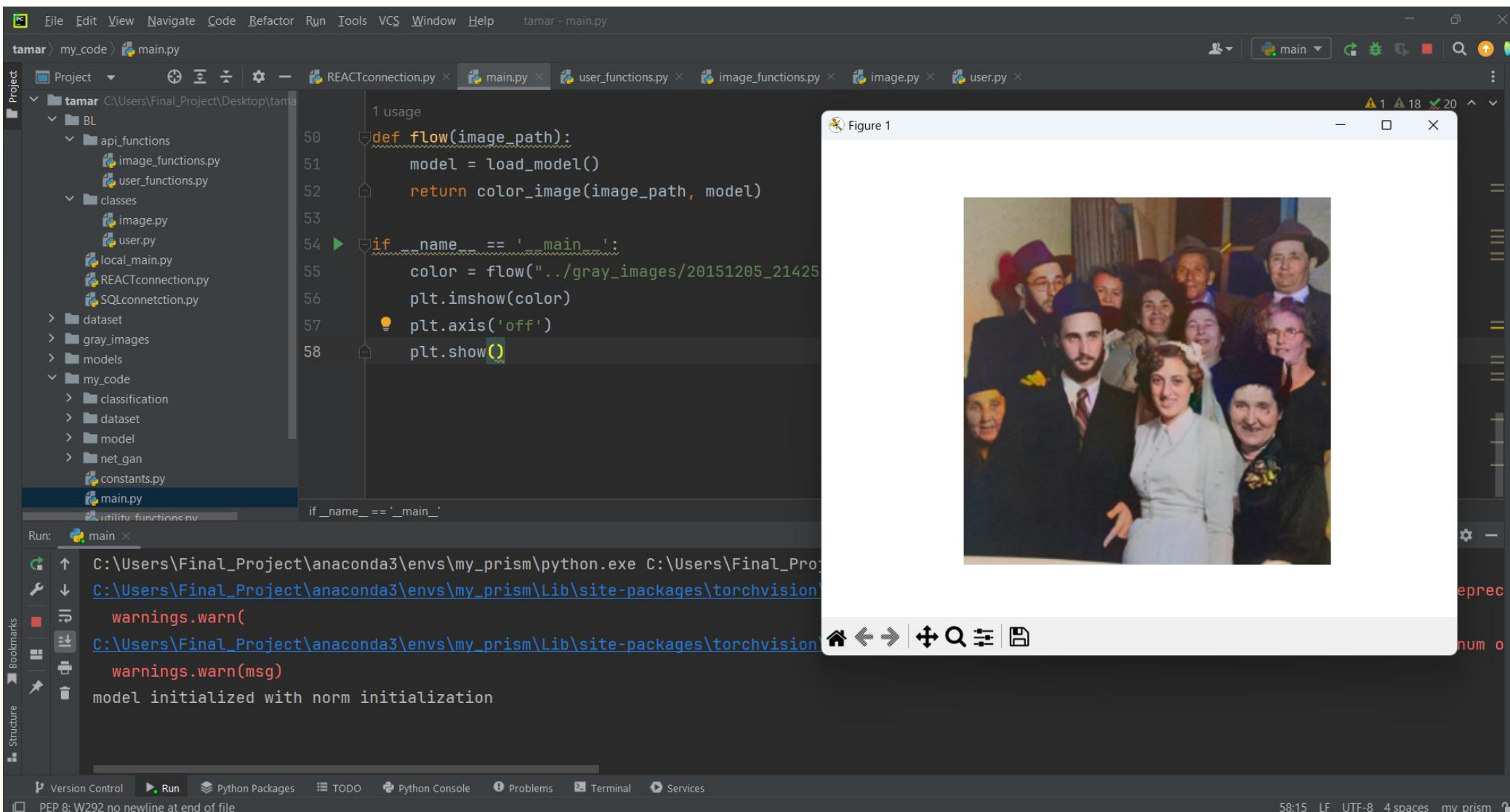
Tamar Gertner

האתגר האלגוריתמי

צביעת תמונות בשחור ולבן - לצבעוניות



הדגמה



The screenshot displays the PyCharm IDE interface. The main editor window shows a Python file named `main.py` with the following code:

```
1 usage
50 def flow(image_path):
51     model = load_model()
52     return color_image(image_path, model)
53
54 if __name__ == '__main__':
55     color = flow("../gray_images/20151205_21425")
56     plt.imshow(color)
57     plt.axis('off')
58     plt.show()
```

The left sidebar shows the project structure, including folders like `api_functions`, `classes`, `dataset`, `gray_images`, `models`, and `my_code`. The bottom status bar indicates the file encoding is UTF-8 and the line length is 58:15.

Overlaid on the right side of the IDE is a window titled "Figure 1" which displays a color photograph of a group of approximately ten people, including men and women of various ages, dressed in formal attire. The photo appears to be a historical or family portrait.

VGG VS. GAN



VGG

מסווג את האובייקטים ולומד את הקשר בין צורה לצבע

פשוט לפיתוח
צריכת עיבוד נמוכה

לא מצליח להפיק תוצאות אמינות מספיק



GAN

מורכב משתי רשתות – מחולל ומאפיין, המתחרות במשחק סכום אפס, ומתקנות אחת את השנייה, עד לקבלת תוצאה מספקת.

מפיק תוצאות מצוינות – אמינות, מגוונות ואחידות.

מורכב יותר לפיתוח
צריכת עיבוד גבוהה





C-GAN

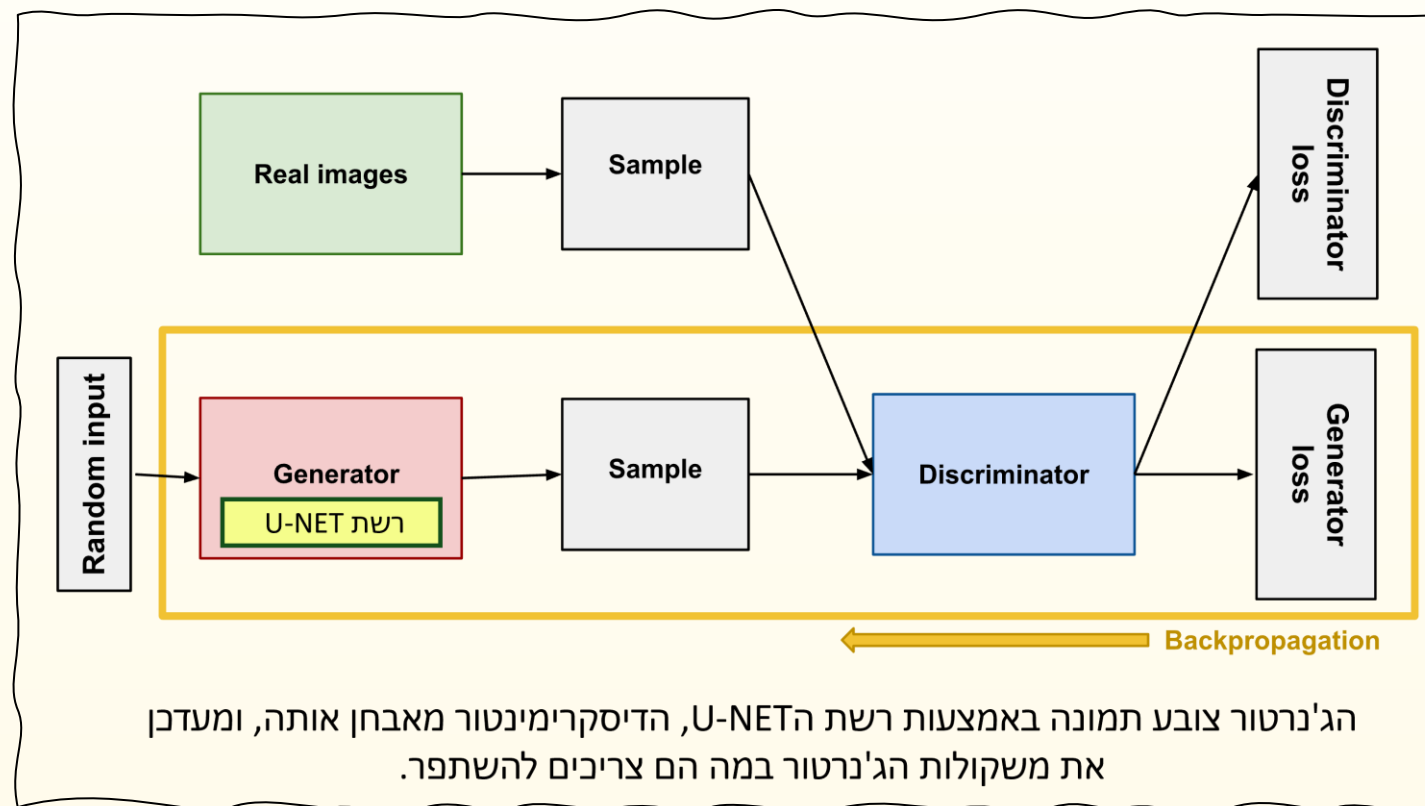
conditional generative adversarial networks



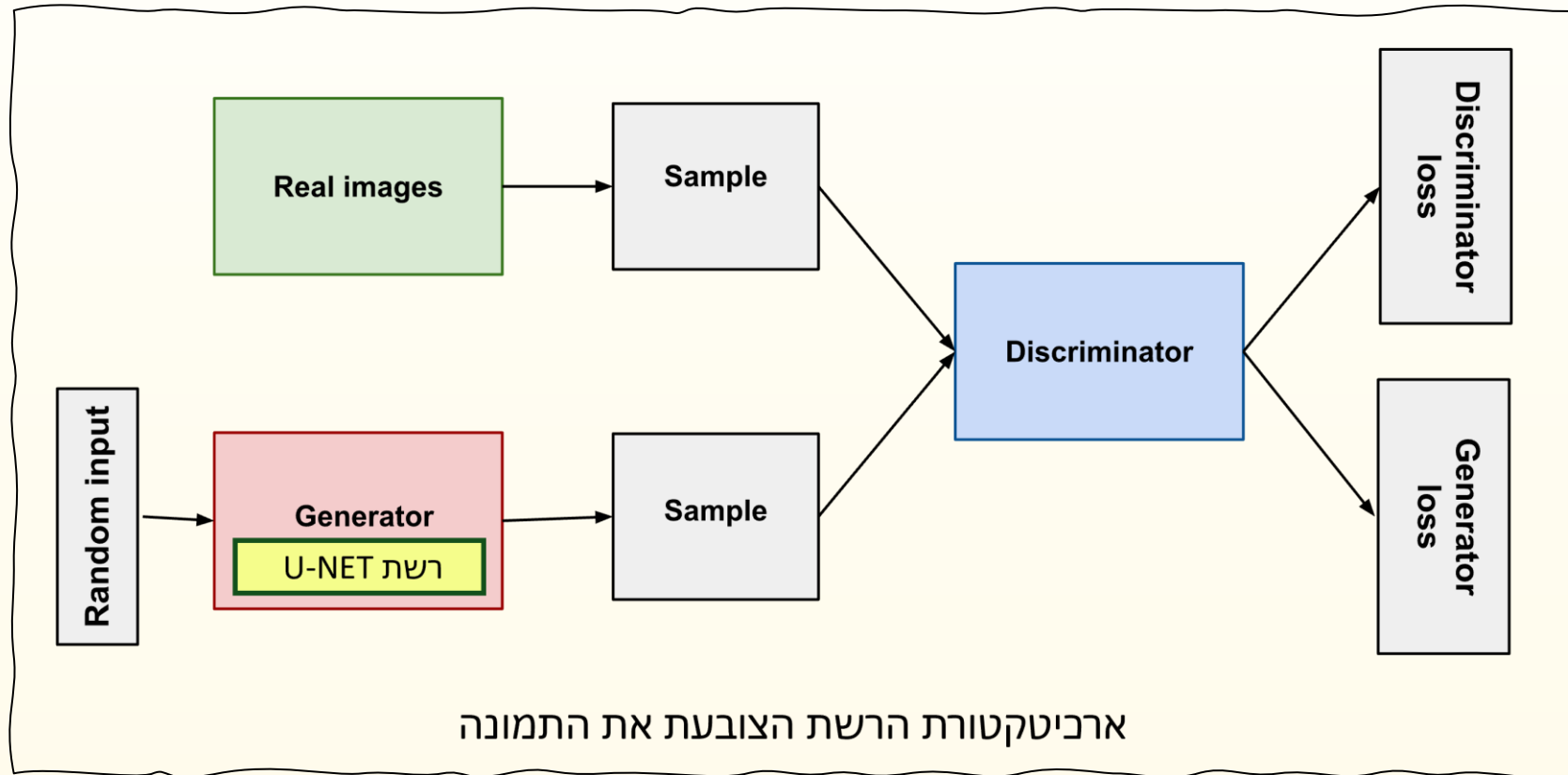
מבנה המודל

רשת GAN מותנית

ובתוכה רשת U-net הנעזרת בתוויות ResNet18

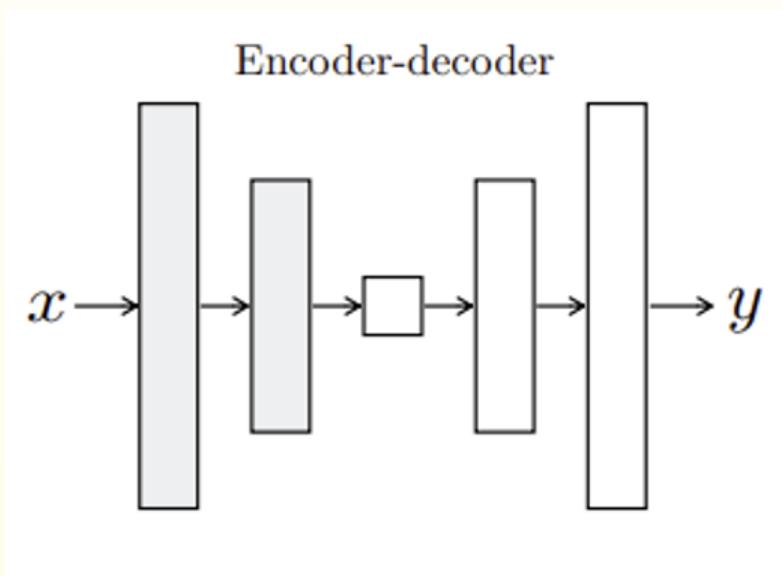


Generator

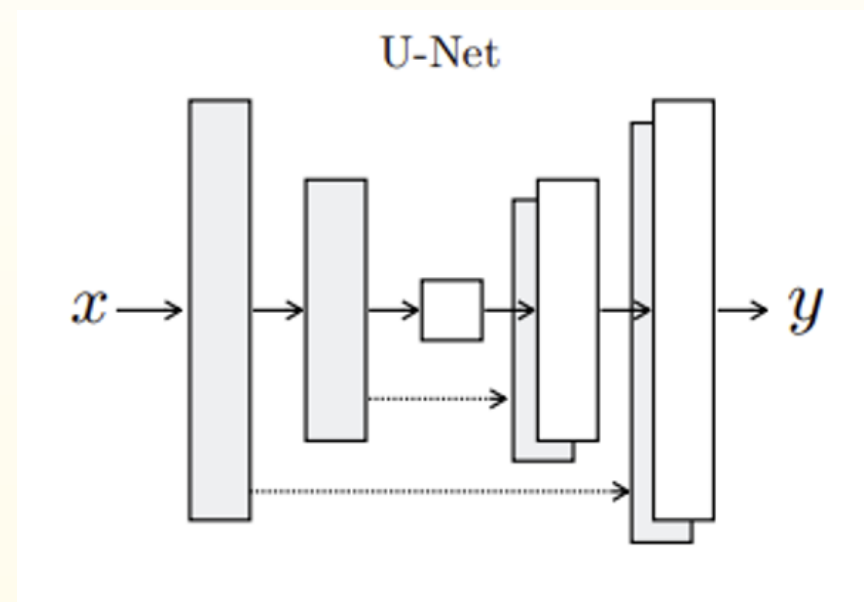


Pix2pix - מבנה הג'נרטור

Unet. VS. מקודד-מפענח:



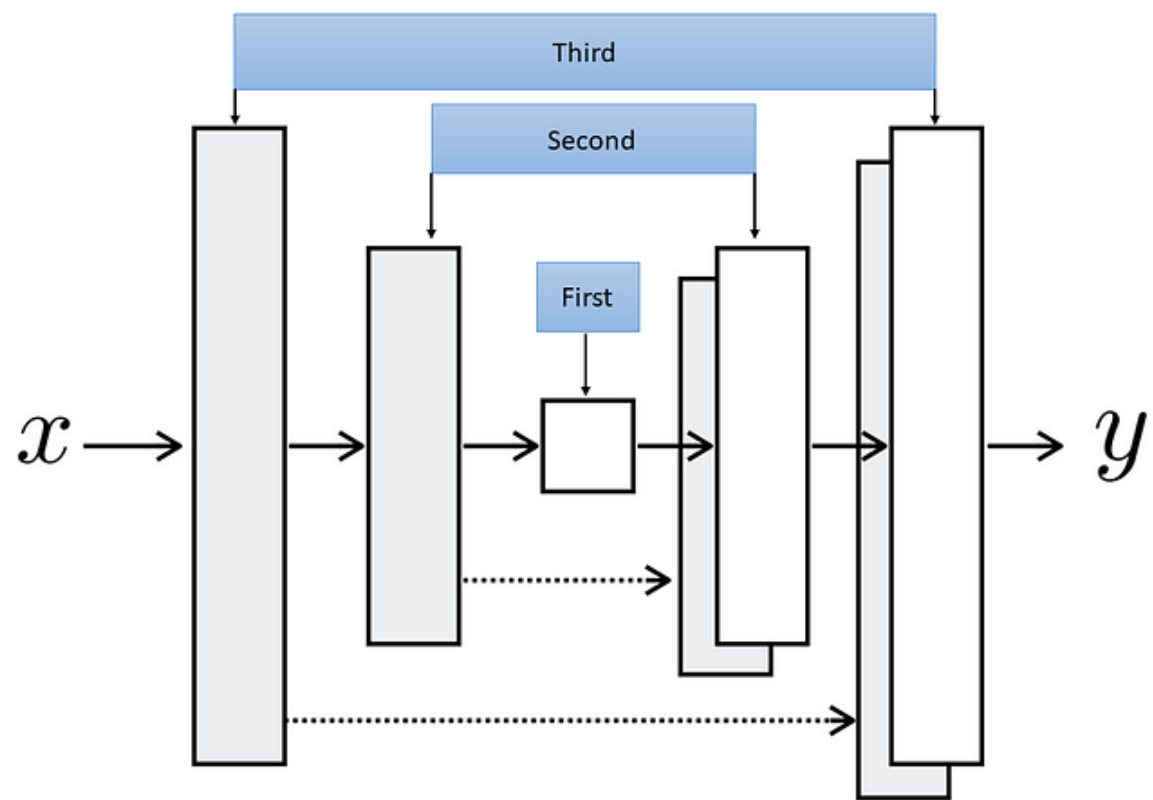
מעביר פרטים גדולים
מנתח בצורה כללית



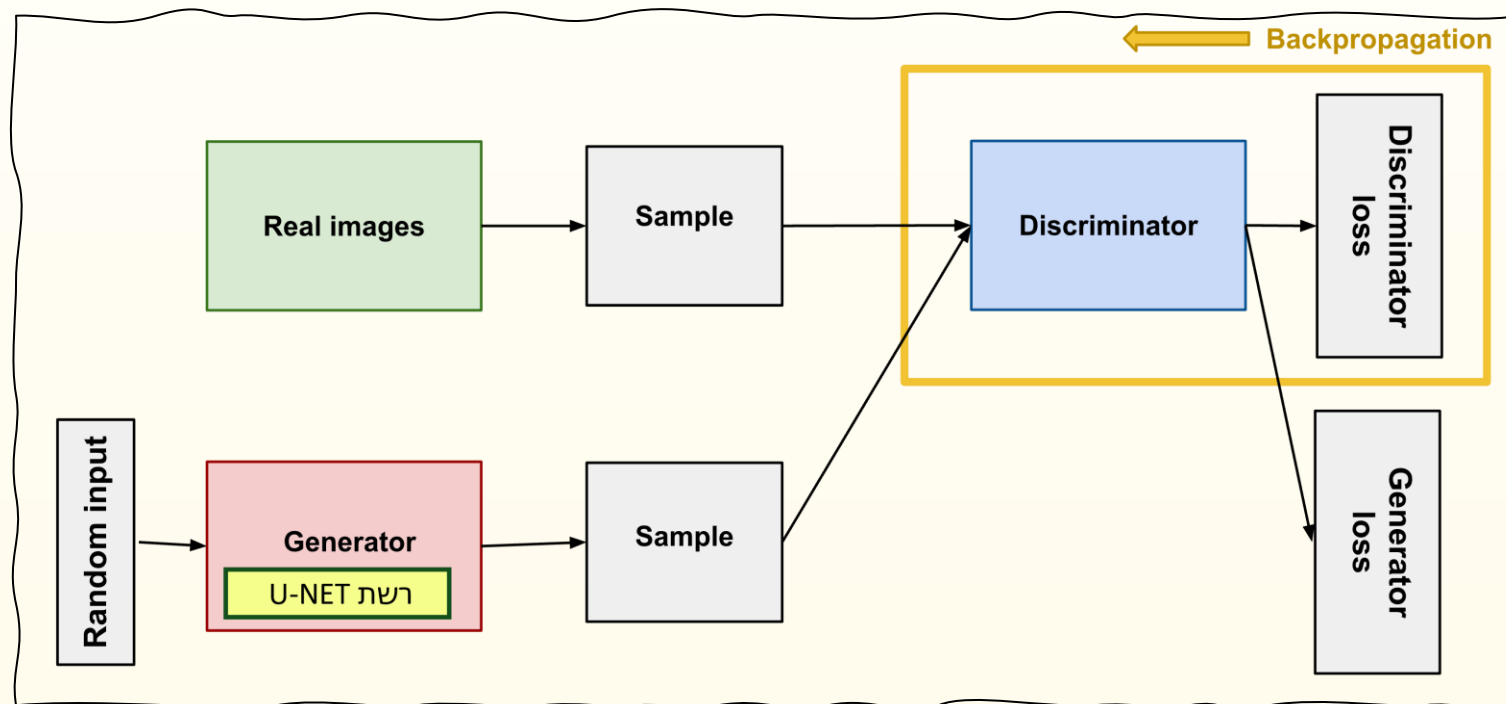
מעביר פיקסל לפיקסל
מנתח מדויק יותר

בניית רשת ה - U-net

רשת ה U-net
שדרת ה ResNet18



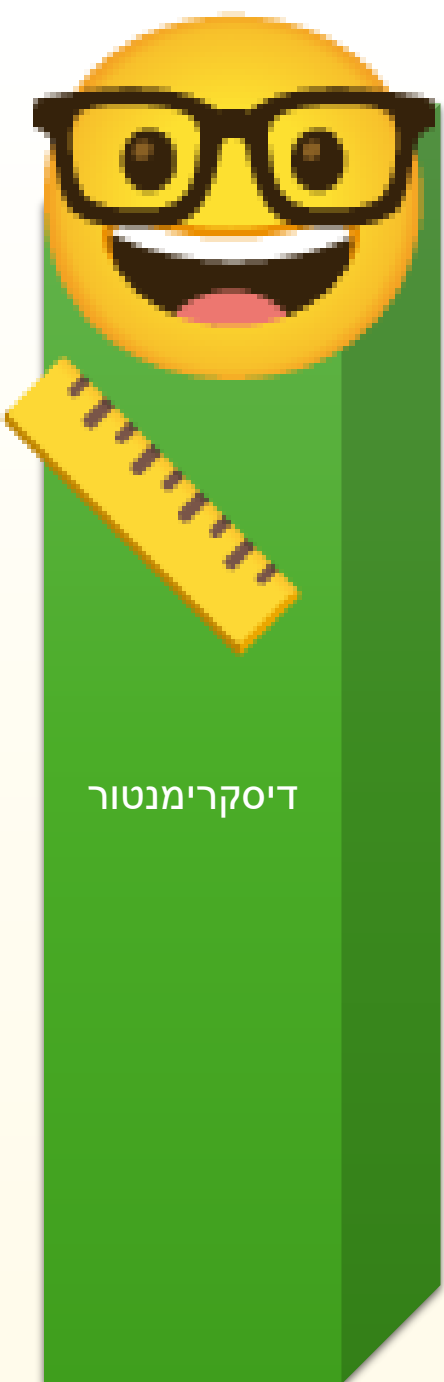
Patch Discriminator





משחק סכום ה-0 התחרותי
של הג'נרטור והדיסקרימינטור

חלחול לאחר של הדיסקרימנטור



הגנרטור חוזר תמונה ומוציא פרדיקציה בשם `fake_image`

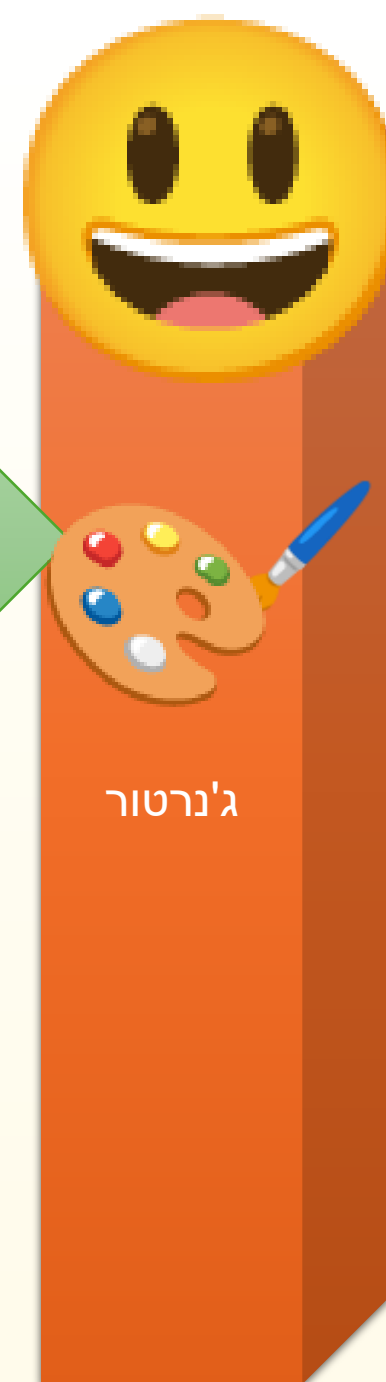
הדיסקרימנטור מנסה לחזות האם היא אמיתית ומוציא את תחזיתו בשם `fake_preds`

הדיסקרימנטור מחשב את פונקציית המחיר ע"פ `fake_preds`

הדיסקרימנטור מקבל תמונה מקורית - ומנסה לחזות האם היא אמיתית ומוציא את תחזיתו בשם `real_preds`

הדיסקרימנטור מחשב את פונקציית המחיר ע"פ `real_preds`

הדיסקרימנטור מחשב את הממוצע בין `real_preds` ל-`fake_preds` ומכוון לעצמו את משקולותיו לפיו



חלחול לאחר של הדיסקרימיננטור

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y} [\log D(x, y)] + \mathbb{E}_{x,z} [\log(1 - D(x, G(x, z)))]$$



חלחול לאחור של הג'נרטור

הגנרטור חוזר תמונה ומוציא פרדיקציה בשם `fake_image`

הדיסקרימינטור מנסה לחזות האם היא אמיתית ומוציא את תחזיתו בשם `fake_preds`

הג'נרטור מחשב את פונקצית המחיר ע"פ `fake_preds` שהורה לו הדיסקרימינטור -
בפונקציות `loss_G_GAN + loss_G_L1`

הג'נרטור מבצע חלחול אחורה לאיפטום המשקולות, ומעדכן את משקולותיו

דיסקרימינטור

ג'נרטור

חלחול לאחר של הג'נרטור

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z} [\|y - G(x, z)\|_1]$$



הג'נרטור הטוב ביותר – בסוף האימון

$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$$



אתגרי ארכיטקטורה ודאטה

חקר מורכב והחלטות ארכיטקטורה.

שינוי מ VGG ל - GAN, וחקר מחודש.

חיפוש דאטה וכתיבת סקריפט להמרתה לשחור לבן



אתגרי אימון

העלאת דאטה-סט בגודל 25 ג'יגה לgoogle cloud והתממשקות עם colab
ו - Kaggle.

אימון כבד.

