

## Stochastic Gradient descent

### MNIST digit classification

#### מטרת הפרויקט:

בנייה רשת נוירונים המסוגלת לזהות בסבירות גבוהה סירה הכתובה בכתב יד שמתקבלת כקלט.

#### הקדמה והגדרות:

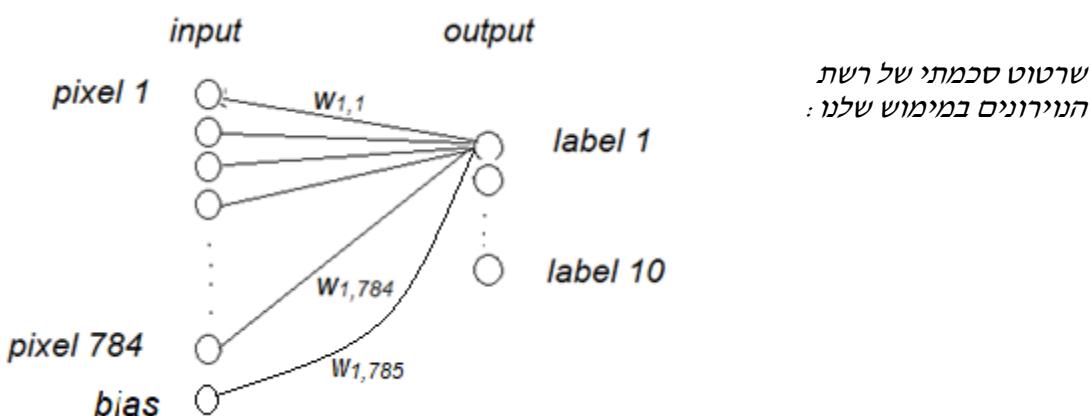
בכדי לישם את המטריה נשתמש במאגר הנתונים MNIST, הכולל 60,000 תמונות מתיוגות של ספרות בין 0 ל-9.

על מנת ללמד את מאגר הנתונים באופן יעיל ונכון, נגידר פונקציית מטריה המתארת את השגיאה של הרשת עבור מאגר הנתונים הנוכחי. נרצה לאמן את הרשות על ידי מציאת נקודת מינימום בשיטה איטרטיבית לפונקציית המטריה, ככלומר מציאת מינימום שגיאה (או קירוב מספק למינימום השגיאה).

לשם כך נמשח את האלגוריתם Stochastic Gradient Descent(SGD)

נגידר :

- $X \in R^{n*m}$  set of  $m$  images, each with size  $n$  pixels.
- labeled data  $\{x_i, y_i\}_{i=1}^m, y_i \in \{1 \dots l\}, x_i \in R^n$ .
- $F(W)$  is the objective function.  $W \in R^{l*n}$ .
- $w_i \in R^n, i \in \{1 \dots l\}$ ,  
is a vector of unknown weights that we will find  
by minimizing the objective function. i.e by calculating  
 $\operatorname{argmin}_{W \in R^{l*n}} F(W)$
- Indicator:  $I_{k=j}$  equals 1 if  $k = j$ , otherwise 0.
- $C_j = [I_{y1=j} \dots I_{ym=j}]^T, j \in \{1 \dots l\}$
- $\alpha$  – learning rate



### הגדרת פונקציית המטרה softmax:

ישנן אפשרויות שונות לבחירת פונקציית המטרה, בבעיית classification שלנו נבחר במודל logistic regression.

פונקציית המטרה softmax מותאמת את מודל השגיאה של logistic regression לבעיה שלנו, המכילה 10 תוויות אפשריות לתמונה.

הפונקציה מתארת את השגיאה של הרשות עבור מאגר הנתונים הנלמד, על סמך W - המשקלות ברשות שלנו :

$$L_X(\{w_k\})_{k=1}^{10} = - \sum_k c_k^T * \log\left(\frac{\exp(X^T w_k)}{\sum_j \exp(X^T w_j)}\right)$$

אתחול המשקלות נעשה על ידי מתן ערכים רנדומליים בטוחה (0,1).

### הגדרת הגרדיינט של פונקציית המטרה:

נגורור את פונקציית המטרה לפי וקטור המשקלות  $W_p$ . נקבל :

$$\nabla_{w_p} L_X = X \left[ \sum_j \frac{\exp(X^T w_p)}{\sum_j \exp(X^T w_j)} - c_p \right]$$

בנוסף, נגדיר וקטור  $b \in R^{10}$ : bias, כך שיחסוב ערך השגיאה יכלול פרמטר חופשי. בימוש שלנו נוסיף פיקסל עם ערך 1 לכל תמונה שנלמד.

### אלגוריתם SGD :

אלגוריתם SGD בוחר בכל איטרציה תות - קבוצה אקראית מתוך מאגר הנתונים ( $X_s \subseteq X$ ), ומחשב גרדיינט עבור פונקציית המטרה המתארת את השגיאה של הקבוצה האקראית שנבחרה. גרדיינט זה ( $\nabla L_{X_s}$ ) מהווה הערכה לנגורור של פונקציית המטרה המקורי, ומאפשר חישוב מהיר יותר של הנגוררות. מכיוון שאנו מחפשים נקודת מינימום, בכל איטרציה נעדכן את משקלות הרשות בכיוון החපוך לגרדיינט.

הערה : פונקציית המטרה שהציגו היא סכום של השגיאות של כל התמונות במאגר הנתונים X. את השגיאה זו נחלק בגודל מאגר הנתונים, וכך נתקיים לפונקציה המחשבת שגיאה עבור תת קבוצה אקראית  $X_s$ , נחלק את השגיאה בגודל תת הקבוצה ונקבל הערכה לשגיאה המקורי על סמך הקבוצה האקראית.

בסיום האלגוריתם אנו מבצעים שתי בדיקות :

- 1- בדיקת אחוזי הצלחה בפענוח של מאגר הנתונים שלנו, על מנת להעריך את הלמידה שביבינו.
- 2- בדיקת אחוזי הצלחה בפענוח של מאגר נתונים שלא למדנו. נצפה לראות אחוזי הצלחה דומים וגובהים בשתי הבדיקות.

### הלואה הראשית במימוש שלנו באופן סכמטי:

Load and shuffle MNIST training data X.

Add bias: add 1 pixel with value 1 to each image.

Initiate  $W \in R^{10 \times 785}$  with random values in range [0,1).

for k=1...maxIter:

calculate loss:  $L_X(W)$  ##to show improvement and convergence

$\mathbb{S} \leftarrow$  list of B random batches

for each s in  $\mathbb{S}$ :

$$F'(W) \leftarrow \frac{1}{|\mathbb{S}|} \nabla L_{X_s}$$

$$W \leftarrow W - \alpha * F'(W) \quad \text{##}\alpha \text{ is the learning rate}$$

classifyImages(training data X) ##checking success rate on the training data

classifyImages(testing data)##checking success rate on the testing data

הערה : מכיוון ש  $x^T w$  לא חסום, חישובי האקספוננט עלולים להיות לא בטוחים. לכן הגדרנו  
 $\eta = \max_j \{x^T w_j\}$

:numerical overflow and underflow בעчисוב האקספוננט החסרנו ערך זה ע"מ להבטיח חישוב בטוח ולמנוע

$$\frac{\exp(x^T w_k)}{\sum_j \exp(x^T w_j)} = \frac{\exp(x^T w_k - \eta)}{\sum_j \exp(x^T w_j - \eta)}$$

בנוסף, ערכי הפיקסלים המקוריים הם בטוחה [0,255]. לאחר טעינת התמונות אלו מבצעים להם נרמול ומחלקיים אותם ב255 לערבים בטוחה [0,1].

### בדיקות הגרדיינט:

מיימשנו את בדיקת הגרדייאנט המתוואר בקובץ NOA.pdf, על מנת לוודא את נכונות חישוב הגרדייאנט שלנו לפונקציית המטרה.

ראשית בחרנו וקבעו רנדומלי  $d$  בגודל 785- כגודל תמונה, וערך  $\epsilon_0$  התחלתי רנדומלי.

השוינו במספר איטרציות בין הערכים המתקבלים מהמשוואות:

$$|f(x + \epsilon_i d) - f(x)| = O(\epsilon_i)$$

$$|f(x + \epsilon_i d) - f(x) - \epsilon_i d^T \text{grad}(x)| = O(\epsilon_i^2),$$

$$\epsilon_i = (0.5)^i \epsilon_0$$

קיבלונו כאמור שהערכים יורדים בקצבו של חצי ורביע בהתאם.

### תוצאות:

נציג את תוצאות האלגוריתם עבור ריצה של 100 איטרציות, כאשר בכל איטרציה מבצעים  $B=30$  shiforim עבור 30 batches.

כיוון שגודל batch יחיד מושפע על איכות ההערכתה של הגרדיינט שאנו מחשבים לגרדיינט פונקציית המטרה (המוחשבת בכל מאגר הנתונים), נציג תוצאות עבור גdzi batch שונים:

Batch Size	Training data success rate	Testing data success rate
1	77.7%	77.8%
5	83.9%	84.0%
10	90.0%	90.0%
20	90.4%	90.2%
50	91.3%	91.3%
100	91.5%	91.6%

כמובן שניitan ניתן לציין גם תוצאות ריצות ארוכות יותר וקומבינציות שונות של הפרמטרים השונים, אך בסיכון הבדיקות השונות מגיעות לאחיזה הצלחה מקסימלי של כ- 91% עבור testing data ובעור Training data.

### מצורפים קבצי התכנית:

.bias – טעינת Data, נרמול הפיקסלים והוספת פיקסל.

.SGD – מימוש Main.py

gradientTest.py – מימוש בדיקת הגרדיינט.

\*הקוד נכתב ונבדק ב Python 3.6.4 :: Anaconda, Inc