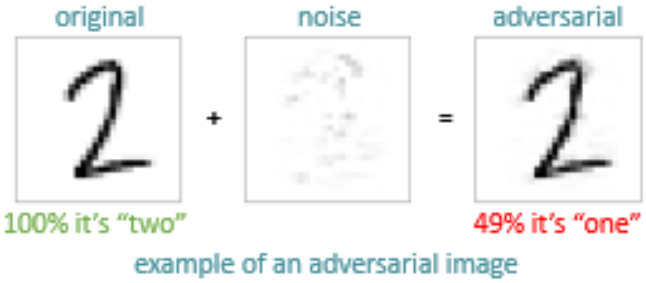
בס"ד

דו"ח מסכם לפרוייקט

סטיב גוטפרוינד וישי אשר

1. תיאור כללי של הפרוייקט ותוצריו

עקבנו אחרי המאמר
"Bridging machine learning and cryptography in defence against adversarial attacks"[1]
המתאר שיטה להגנה על מודלים של Deep neural network מפני adversarial attacks.
הטענה המרכזית שלהם הייתה שאם מפעילים פרמוטציה רנדומית (הכוונה סודית בהקשר הזה) על תמונה בדרכה למודל DNN , המודל מצליח ללמוד – ואחוזי הצלחת התקיפה יורדים משמעותית .
המטרה הכללית שלנו בפרוייקט הייתה לשחזר את המאמר ולאמת או להפריך את תוצאות כותבי המאמר .
כלומר לראות האם ניתן ללמד מודל בצורה מוצלחת שמצליח להגן נגד תקיפות מהסוג המוזכר לעיל .
להלן דוגמא לadversarial example :



השלבים באופן כללי :

א. יצרנו את המודלים של DNN המתוארים במאמר על הdatasets המתוארים במאמר (MNIST , FASHION-MNIST) – והגענו לאחוזי דיוק גבוהים כמצופה .

ב. יצירת מודלים שמעבירים את הקלט שלהם , כלומר תמונה , דרך אחד ממנגנוני ההצפנה הבאים (לפני שעוברת התמונה ללמידה במודל) :
(1) פרמטוציה רנדומית של הפיקסלים (כלומר הBytes) .
(2) AES in ECB mode of operation.
(3) AES in CBC mode of operation.
(4) AES in CTR mode of operation.

לגבי (1) : קיבלנו אחוזים גבוהים (לא כמו ללא פרמוטציה אך מספיק גבוהים) כמו המוצג מאמר .

לגבי (2) : קיבלנו אחוזים לא גבוהים כמו א אך לא ממש נמוכים – זה נשמר לעבודה עתידית .

[1] https://arxiv.org/pdf/1809.01715

לגבי (3): קיבלנו תוצאות דיוק ממש נמוכות ששקולות להגרלת הסיווג . (מבין 10 מחלקות שיש בMNIST ובFASHION-MNIST להגריל את המחלקה הנכונה זה בהסתברות 10% שזה אחוז הדיוק שקיבלנו) .

לגבי (4) : קיבלנו אחוזים דיוק גבוהים .

פירוט על הגרסאות השונות בחלק הבא .

ג. כתוצאה מהשלב הקודם , עברנו לתקוף (בהתקפות FGSM,CW) את המודלים השונים בגירסאות השונות (ללא הצפנה , או עם הצפנת AES-CTR או עם הפעלת פרמוטציה[2]) :
כאשר אין הפעלה של פרמוטציה על הקלטים ההתקפות צלחו כמצופה .
כאשר יש הפעלת פרמוטציה – בחלק זה מימשנו את ההתקפות בשתי גירסאות כי לא הצלחנו להגיע לאחוזים של כותבי המאמר , בגירסה הראשונה ההתקפות צלחו כמעט כמו ללא הפעלת פרמוטציה (כלומר, בניגוד לטענת כותבי המאמר) . ולעומת זאת גירסה השנייה אחוזי הצלחת ההתקפה ירדו , כלומר הגענו (בסדרי גודל) לאחוזים המתוארים במאמר . כלומר הצלחנו לאמת את תוצאותיו .
בהצפנת AES-CTR תקפנו בגירסה השנייה ואחוזי ההתקפה ירדו משמעותית .

פירוט על הגרסאות השונות בחלק הבא .

ד. במקביל לחלק מהתקפות בדקנו נושא מעניין לכשעצמו :
איך בדיוק מודל DNN מצליח ללמוד כאשר מופעלת פרמוטציה ? הרי פרמוטציה רנדומית אמורה להגן על הDATA ...
רצינו לבדוק האם הסיבה שהמודל מצליח ללמוד היא שהתמונות בגודל קטן מדי ועל כן הפיקסלים המשמעותיים צפופים מספיק בשביל ללמוד תבנית מסויימת. על כן ניסינו לעשות ריפוד (PADDING) של התמונות בפיקסלים לבנים (ערך 0) ו**עדיין** המודל הצליח ללמוד . כלומר המודל מצליח ללמוד גם כאשר יש פיזור רחב של הפיקסלים המשמעותיים על פני תמונה גדולה.
נרצה בעבודה עתידית להמשיך לחקור בנושא הזה .

[2] שאר ההצפנות לא עניינו בגלל אחוזי הדיוק של המודלים המתאימים להם או שנשמרו לעבודה עתידית.

2. תיאור מפורט של כל מרכיביו (כולל מודל/ארכיטקטורה/תרשימים וכו') כולל אתגרים ופתרונות

א. יצירת המודלים

בהתחלה כתבנו את המודלים של DNN המתוארים במאמר :

Table 1. Architecture of the DNN classifiers used for the FGSM attack.

Layer	# filters	kernel size
Conv2D	64	8×8
ReLu		
Conv2D	128	6×6
ReLu		
Conv2D	128	5×5
ReLu		
Flatten		
Dense		10

Table 2. Architecture of the DNN classifiers used for the CW attack.

Layer	# filters	kernel size
Conv2D	32/64	3×3
ReLu		
Conv2D	32/64	3×3
ReLu		
MaxPool		2×2
Conv2D	64/128	3×3
ReLu		
Conv2D	64/128	3×3
ReLu		
MaxPool		2×2
Flatten		
Dense		200/256
ReLu		
Dropout		
Dense		200/256
ReLu		
Dense		10

כאשר:

ב2 Table , מתוארים שני מודלים (בטבלה מתוארים בחלק מן השדות x/y : x על פני כל הטבלה שייך למודל הראשון , וy על פני כל הטבלה שייך למודל השני) . המודל הראשון בשאר הדו"חות נקרא מודל CW1 , והשני נקרא CW2 , שוב על שם ההתקפה . אמנם בפועל במודל CW2 ניסינו לעשות שימוש לאימון על הdataset : CIFAR-10 , אמנם האחוזים שקיבלנו לא היו גבוהים מספיק כדי שזה יעניין וכן במאמר CIFAR-10 לא מוזכר בפירוש אלא יש רק שימוש במודל הראשון .[3]
על כן , רק CW1 מעניין אותנו , נקרא לו לשם הפשטות מודל A .

[3] הסיבה שהם משתמשים ספציפית במודלים הללו , היא שבמודלים הללו השתמשו במאמרים המקוריים של ההתקפות FGSM וCW , והם רצו להיצמד למקור . אמנם בפועל לא היה שימוש במודל השני . (שנקרא CW2)
וכן הם אומרים שהשתמש בקוד המקורי של ההתקפות שנמצא בgithub . אנחנו באופן דומה השתמשנו בהתקפות עם שינויים משלנו .

בס"ד

ב1 Table , מתואר המודל שמשתמשים בו בהתקפת FGSM – המודל הזה בשאר הדו"חות נקרא מודל FGSM על שם ההתקפה , נקרא לו לשם הפשטות מודל B .
את המודלים הנ"ל הרצנו על הdatasets הבאים : MNIST, FASHION-MNIST .
מימשנו את המודלים הנ"ל בpython תוך שימוש בספריית tensorflow וכן בkeras .
הגענו לאחוזי דיוק גבוהים כמו של המאמר , כמפורט לעיל בטבלאות .
נציין כי בטבלאות מה שמוצג הוא <u>אחוזי שגיאה</u> .
כמו כן , הדיוק שאנו בודקים הוא מול 1000 התמונות הראשונות שבtest set המתאים לכל dataset . (**בכל הטבלאות**)

ללא פרמוטציה

אחוזי השגיאה שלנו

	mnist	fashion_mnist
Model A	1.5	8.3
Model B	2.1	9.5

אחוזי השגיאה של המאמר

	mnist	fashion_mnist
Model A	1	7.5
Model B	1	8.6

כפי שניתן לראות אכן האחוזים דומים מאוד , והם אחוזי שגיאה נמוכים למדי (= אחוזי דיוק גבוהים)

ב. <u>הצפנות</u>

אימנו את המודלים A,B המתוארים לעיל – רק שלכל מודל הוספנו שכבה לפני האימון שבה מעבירים את התמונות דרך שינוי מסויים :

(1) פרמטוציה רנדומית של הפיקסלים (כלומר הBytes) .
(2) AES in ECB mode of operation.
(3) AES in CBC mode of operation.
(4) AES in CTR mode of operation.

שיטה (1) היא השיטה המוצגת במאמר ואכן הגענו לאחוזי דיוק גבוהים כשל המאמר :

עם פרמוטציה

אחוזי השגיאה שלנו

	mnist	fashion_mnist
Model A	3.7	12.3
Model B	4.2	12

אחוזי השגיאה של המאמר

	mnist	fashion_mnist
Model A	3	11.5
Model B	1.4	11.2

שיטות (4)-(2) הן הצפנת AES במוד ההפעלה המתאים כאשר <u>חלק זה לא הוזכר במאמר</u> כלומר עשינו אותו כתוספת כדי לבחון שיטות הצפנה שונות.

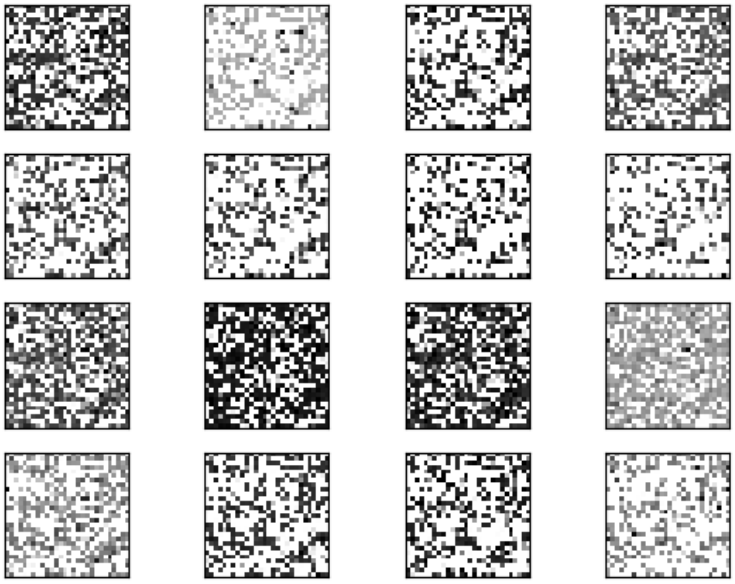
בס"ד

ראשית נראה דגימה של תמונות והפעלת פרמוטציה עליהם (ניתן להראות זאת גם עם הפעלת AES עם כל mode of operation שתואר לעיל) , זאת על תמונות מFASHION-MNIST .

כאשר מוצגות 4 מחלקות ו4 דוגמאות מכל מחלקה , ובהתאמה את הפעלת הפרמוטציה עליה .

original

permutated



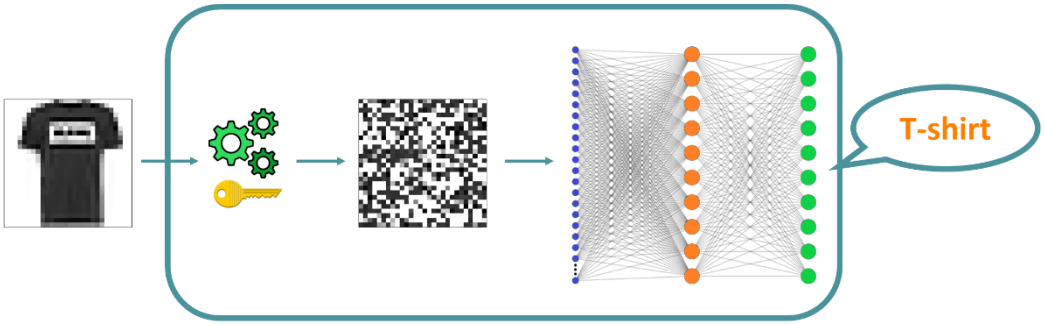
התוצאות היו מפתיעות :

מצד אחד עבור CBC קיבלנו אחוזי דיוק נמוכים כמצופה שכן הצפנה לא אמורה להשאיר עקבות לתבנית מסוימת שבתמונה ,וכן אם התמונה שונה הפלט [החל מהשוני] יכול להיות שונה לגמרי ואין דרך ללמוד כלום. מצד שני עבור CTR קיבלנו אחוזי דיוק גבוהים , ומצד שלישי בECB קיבלנו אחוזים שאינם מספקים אך לא נמוכים מאוד כמו CBC .

אנחנו מבינים כי ההסבר לכך שבCTR קיבלנו אחוזי דיוק גבוהים הוא שבCTR מכניסים את כל התמונות שנכנסות ללמידה במודל , אחרי שמפעילים עליהם xor עם אותה "מסכה אקראית" (האקראיות של המסכה היא מתקבלת מהמפתח ומהAES - block cipher) ומכיוון שמדובר באותה מסיכה אזי עבור בלוקים זהים במיקומים זהים נקבל את אותה הצפנה , לכן יישמר המבנה המקורי ברוב התמונה , והמודל מצליח ללמוד .

היינו מצפים שבECB גם נקבל אחוזי דיוק גבוהים כשל CTR (שכן בECB בלוקים זהים מוצפנים לאותם בלוקים). הסבר אפשרי יהיה שאמנם בלוקים זהים מוצפנים לאותם בלוקים אבל אין התייחסות למיקום ולכן חלק ממבנה התמונה לא נשמר. נבחון את זה ונבדוק אפשרויות שונות ב future work.

מצ"ב הארכיטקטורה עבור שיטות ההצפנה[4] :



הארכיטקטורה של מודלים מצפינים

נציין שבהתחלה הגענו לאחוזי דיוק נמוכים בכל שיטות ההצפנה עם AES .[5]

לכן כדי שנמשיך לנסות לתקוף את המודלים שמשתמשים בשיטות הללו , ניסינו להצפין בצורה שונה .

בהתחלה שיטת ההצפנה שבה הצפנו היא :

לקבל תמונה (בגודל 28 X28 bytes) להשטיח אותה לווקטור בגודל 784 (בעצם רצף של שורות המטריצה של התמונה), להצפין את הווקטור הנ"ל (כמובן בלוק-בלוק לפי שיטת ההצפנה AES בmodes השונים) , לאחר מכן קיבלנו ווקטור חדש (מוצפן) בגודל 784 , ואז החזרנו אותו למימדים המקוריים 28X28 . נסמן שיטה זו ב flattening (האחוזים בהתאמה יהיו בטבלה)

לאחר נכן כשקיבלנו אחוזים לא טובים ניסינו את השיטה הבאה :

---

[4] הדוגמא כאן היא תמונה שעוברת בפרמוטציה

[5] מלבד ECB.

לקבל תמונה (בגודל 28 X28 bytes) לפצל אותה לבלוקים של 4X4 (החל מפינה השמאלית העליונה), לשטח כל בלוק לווקטור בגודל 16 , ואז לאחד את הבלוקים לווקטור בגודל 784 להצפין את הווקטור הנ"ל (כמובן בלוק-בלוק לפי שיטת ההצפנה AES בmodes השונים) , לאחר מכן קיבלנו את ווקטור חדש (מוצפן) בגודל 784 , ואז החזרנו אותו למימדים המקוריים 28X28 . (באותו אופן החזרת הבלוקים למקומם) .

גם בשיטה זו קיבלנו אחוזי דיוק נמוכים .

בשלב מאוחר יותר - בו וידאנו את התוצאות התברר כי היה באג , ולאחר שינוי המימוש[6] , קיבלנו כי בCTR mode of encryption אחוזי הדיוק מספקים למדי , כנ"ל .

להלן טבלת אחוזי השגיאה עבור AES :

AES		Dataset	
mode	model	mnist	fashion mnist
ECB	A	18.40	54.60
	B	19.30	55.30
CBC	A	67.60	71.50
	B	87.40	90.30
CTR	A	3.70	17.40
	B	2.70	16.70

ג. <u>התקפות</u>

בשלב זה תקפנו את המודלים A וB ללא הפעלת פרמוטציה על הdatasets המוזכרים לעיל .
ואכן קיבלנו אחוזי הצלחה גבוהים של התקפה כמצופה . ההתקפות שהתקפנו באמצעותן הן :
FGSM וCW כאשר בCW ההתקפה תלויה באיזו נורמה בוחרים : $l_0/\ l_2/\ l_\infty$
כאשר התקפת CW משתמשת בנורמה (פונ' מרחק) כדי להעריך loss .[7]

מצ"ב תוצאות התקיפה על המודלים *A,B* (ללא הפעלת הפרמוטציה) :

<u>אחוזי השגיאה שלנו:</u>

model	attack type	mnist	fashion_mnist
A	CW l_2	100.00	100.00
	CW l_0	100.00	100.00
	CW l_∞	100.00	100.00
B	FGSM	39.50	77.20

[6] התברר כי לא השתמשנו במפתח קבוע .
[7] כדי ליצור adversarial example : CW בעצם מבצע למידה על בסיס תמונה תוך מיזעור הלכלוך שמנסה להוסיף על מנת להטעות את המודל (או לחילופין יש כישלון בהתקפה)

אחוזי השגיאה של המאמר:

model	attack type	mnist	fashion_mnist
A	CW l_2	100.00	100.00
	CW l_0	100.00	100.00
	CW l_∞	99.99	99.90
B	FGSM	92.10	60.60

הערה :

בהתקפת FGSM ניתן לראות שאחוזי השגיאה שונים בין מודל B אצלינו לבין מודל B במאמר . בהתקפת FGSM ישנם פרמטרים כגון ε שמייצג את רמת הdistortion (עיוות) שאנחנו מרשים. (עיוות מהתמונה המקורית) אמנם פרמטר זה אינו נמצא במאמר . (וכמו שבהמשך נראה במייל לאחת מכותבי המאמר – הם לא חשפו את הפרמטרים השונים) .

אנחנו קבענו $\varepsilon = 0.1$. (קבענו אותו על בסיס הלכלוך שהוא מוסיף – עפ"י היראות כללית)

התקפות על מודלים עם פרמוטציה :

בשלב זה התעסקנו הרבה זמן .

אנחנו הבנו מהמאמר שהתוקף (הן בCW והן בFGSM) מקבל גישה למודל עם פרמוטציה. (בCW צריך לספק לו פונקציה predict שמחזירה את פלט המודל שכבה אחת לפני הsoftmax)

בשלב זה הסתבכנו מבחינת המימוש להכניס את הפרמוטציה להתקפה עצמה , בעיקר בCW .

המחשבה הראשונית הייתה להעביר תמונות שעברו פרמוטציות ואותן לתת לCW לתקוף כאשר הפונ' predict שמקבלת תמונה תחזיר את הסיווג מהמודל שלמד את הפרמוטציה (A עם הפעלת פרמוטציה) . אמנם הבנו כי מודל זה אינו הגיוני , שכן התוקף מקבל את התמונות שעברו פרמוטציה והוא לא לומד את הלכלוך שהוא מוסיף (בנוסף הוא אינו מחזיר תמונה מקורית) , בעוד שהתוקף אמור להיות אדפטיבי , כלומר predict אמור להעביר דרך הפרמוטציה כל תמונה שהתוקף יבקש .

לכן עברנו לנסות לממש את הפונקציה predict כאשר היא **כן** מעבירה לפני כן בפרמוטציה .

בהמשך ניסינו לממש את הכנסת הפרמוטציה ל predict בעזרת numpy (בעזרת פונקציה של פרמוטציה שם ואכן כל המודלים שלנו אומנו בעזרת פרמוטציה שנבנתה בעזרת numpy עם seed = 42)

אמנם היה קשה לעשות את האינטגרציה בין numpy לבין הtensors שההתקפה CW השתמשה בהם . (כי tensorflow עובד עם גרף חישוב , ולכן אי אפשר להפעיל numpy על שום דבר)

לכן עברנו להשתמש בפונקציות מtensorflow (בשלב זה היו הרבה באגים במימוש) .

אמנם הגענו לאחוזי שגיאה בדיוק כמו אחוזי השגיאה של המודל הלא מותקף , כלומר ההתקפה לא עשתה כלום . (בעוד שבמאמר ההתקפה הצליחה משמעותית פחות אבל הצליחה לפחות במקצת , כלומר בשלב זה לא היינו בטוחים האם זה באג או שהגענו בצדק לתוצאות שונות)

בס"ד

בשלב זה, בעצת המנחה שלנו, שלחנו מייל לאחת מכותבי המאמר בבקשה לשתף בפרמטרים של התקיפה ומה האופן בו הם ביצעו את התקיפה .

להלן המייל:

Hello,

As part of our final project for a Bachelor's degree in Computer Science, we are trying to recreate your article "Bridging machine learning and cryptography in defense against adversarial attacks".

We've build the same models as you have described and indeed got to the same accuracies.
We are now trying to implement the two attacks: CW and FGSM.

1. We are struggling a lot with CW. We indeed got 100% error rate on the classical classifier, but we're not managing with the attack on the classifier with permutated data. We are stuck with implementation. We tried many variations, but all failed . The way CW should work while attacking with permutation , as we understand, is to call as many times the 'predict' function of our model as needed for learning. The first thing 'predict' is doing, is permutating the input. At first we tried implementing the permutation with numpy, but this failed because CW is using a computation graph from tensorflow. We then tried permutating with tensorflows functions, ran the model on 1000 examples and got the same error rate as on the original data, i.e. the attack didn't change anything.

   Is it possible for you to share your code doing this process?

2. FGSM seems to work quite fine except that we don't get the same percentages as you describe in the article.

   Can you please share with us the parameters you used for the attack (eps, clip_min and clip_max)

We would be very grateful if you can help us out.

Thank you,
Yishay and Steve

P.S. This mail is approved by our instructor, he's added to the mail as well.

ותגובתה :

Hello,

Unfortunately, due to the trouble with my computer I have not the access to that code now.

But, I can suggest you to read the continuation of the "Bridging machine learning..." --
> http://sip.unige.ch/projects/snf-it-dis/publications/cvpr-2019 This new paper is accepted for CVPR this year.

The Python code, trained models and used attacked and original data are available on
Github https://github.com/taranO/defending-adversarial-attacks-by-RD

Don't hesitate to contact me if you have any questions.

Best regards.

TARAN Olga

בס"ד

היא הפנתה למאמר חדש שלהם בנושא ואמרה שהקוד שלהם שכתבו במסגרת המאמר הזה לא זמין .

המשכנו לדבג ולנסות פונקציות חדשות .

בסופו של דבר מצאנו פונקציה של tensorflow שמקבלת פרמוטציה ויכולה להפעיל אותה על tensors . נתנו לפונקציה הזו את הפרמוטציה המתקבלת מnumpy (עם seed = 42) .

והגענו לאחוזי שגיאה 100% (בהתקפת CW , בFGSM טיפה פחות "טוב") כלומר 100% הצלחה בהתקפה . (הבאג הקודם היה באג מימושי)

אמנם , כל הרעיון של המאמר היה לאמן מודלים עם פרמוטציה סודית , ואז ההתקפה אמורה להצליח משמעותית פחות (ברמת האחוזים הבודדים) .

לאחר מכן בעצת המנחה שלנו , שלחנו מייל לניקולס קרליני (אחד מכותבי ההתקפה CW : C עבור Carlini)

שאלנו אותו האם הדרך בה אנחנו תוקפים עם CW (שהפונקציה predict מעבירה את הקלט שלה בפרמוטציה) היא נכונה , או שאנחנו נותנים אולי יותר מדי כח לתוקף בכך .

להלן המייל :

Hello,

As part of our final project for a Bachelor's degree in Computer Science, we are trying to reproduce the article "Bridging machine learning and cryptography in defense against adversarial attacks".

The article tries to 'defend' models from adversarial examples. One of the attacks used in the article is the CW attack.
The way they try to defend themselves from the attack is by performing a random permutation on every image and letting a model learn from the permutated images.
They claim that by doing so, they successfully decrease the error rate from 100% on the original data to ~8% error rate on the permutated data.

Our results are different.
We get a 100% error rate on the permutated data as well.

We have a few questions we hope you would be so kind to help us:

1. What does it mean when the CW attack fails on some image? What are we expected to get as output from the attack function? (The exact same original image?)

2. The way we implemented the permutations is as follows: whenever we call 'predict' on an image (an original one), our model first permutates this image and then feed to the layers.
Likewise , in the CW attack the input to the "predict" function passes through permutation and the normal prediction is applied to the permutated input. (returns the pre-softmax layer)
Is that the way you think we should do so ?
Is that maybe too much power we give to the attacker ?

Thank you in advance ,

Steve and Yishay .

תגובתו הייתה שאכן זה יותר מדי כח לתוקף .[8]

Hi --

That paper looks like they argue security only in the "grey-box" threat model where they "assume that the attacker does not have direct access to the secret block" but "he completely knows the system architecture".

So you may be violating the threat model by directly feeding this into the attack algorithm. (When you do feed it in, I would definitely expect a 100% attack success rate. Be sure to measure the distortion too, though, to see how that changes.)

However: you might be able to still defeat this defense by applying some form of expectation over the different values of random noise, even if you don't directly feed the specific secret into the defense.

That said, an unbounded attack should never fail -- given sufficient distortion, it should always succeed (if only with extremely large distortion so as to be not be meaningfully called an adversarial example).

Nicholas

אנחנו הבנו שזה נכון כי בעצם התוקף לומד גם את הפרמוטציה שהיא בעצם שכבה ליניארית – מטריצה בגודל 784 על 784 כאשר כל שורה מתאימה לתא בודד בתמונה , באופן הבא:

אם נסתכל במטריצה הזו בשורה ה-i אזי אם יש 1 בתא ה-j זה אומר שצריך לשים את הפיקסל ה-j בתמונה בתא ה-i בתמונה החדשה. (שעברה פרמוטציה) . כמו כן בשורה ה-i כל השורה היא אפסים מלבד התא המתאים.

בעצם מה שאנחנו עשינו עד עכשיו הוא שיטת white-box – התוקף יודע גם את הארכיטקטורה וגם את הפרמטרים של המודל אותו הוא תוקף .

בעוד מה שעשו במאמר היא שיטת grey-box כלומר התוקף יודע את הארכיטקטורה , אבל אינו יודע את הפרמטרים .

בשיטה זו נתנו לתוקף גישה למודל (מסוג A) שאומן ללא פרמוטציה , ואז בדקנו את התמונות מול מודל (מאותו סוג) שמעביר את התמונות פרמוטציה .

[8] כפי שניתן לראות במייל (תחילת פסקה However) הוא מציע הצעה איך בכל זאת להתקיף בהצלחה . זה יישמר לעבודה עתידית .

ואכן בשיטה זו (grey-box) קיבלנו אחוזים דומים לאלו של המאמר :

<u>אחוזי השגיאה שלנו:</u>

model	attack type	mnist	fashion_mnist
A	CW l_2	4.50	12.70
	CW l_0	7.30	12.50
	CW l_∞	5.40	12.90
B	FGSM	8.60	29.80

<u>אחוזי השגיאה של המאמר:</u>

model	attack type	mnist	fashion_mnist
A	CW l_2	8.64	12.12
	CW l_0	14.53	13.48
	CW l_∞	12.24	12.55
B	FGSM	18.00	27.50

אכן ניתן לראות שבשיטה זו אחוזי השגיאה ירדו משמעותית , כלומר ההתקפה נחלשה משמעותית .

בנוסף , בשיטת ההצפנה AES in CTR mode of encryption בשיטת grey-box:

model	attack type	mnist	fashion_mnist
A	CW l_2	4.20	17.20
	CW l_0	9.60	18.70
	CW l_∞	4.90	17.80
B	FGSM	4.90	26.50

כלומר אחוזי הדיוק גבוהים למדי כמו בשיטת הפעלת הפרמוטציה כלומר זו עוד שיטה שעובדת שמצליחה להגן (וכמובן ללמוד בצורה טובה).

בס"ד

נסכם את התוצאות שלנו , להלן **טבלה מסכמת של אחוזי השגיאה** :

MNIST

Model	Images	Unsecured	Permutation	AES · ECB	AES · CBC	AES · CTR
A	Originals	1.50	3.70	18.40	67.60	3.70
	CW l_2	100.00	4.50			4.20
	CW l_0	100.00	7.30			9.60
	CW l_∞	100.00	5.40			4.90
B	Originals	2.10	4.20	19.30	87.40	2.70
	FGSM	39.50	8.60			4.90

FASHION-MNIST

Model	Images	Unsecured	Permutation	AES · ECB	AES · CBC	AES · CTR
A	Originals	8.30	12.30	54.60	71.50	17.40
	CW l_2	100.00	12.70			17.20
	CW l_0	100.00	12.50			18.70
	CW l_∞	100.00	12.90			17.80
B	Originals	9.50	12.00	55.30	90.30	16.70
	FGSM	77.20	29.80			26.50

3. <u>עבודה עתידית</u>

כפי שהזכרנו במהלך הדו"ח , על הדברים הבאים נרצה לעבוד בעתיד ולבדוק :

א. ECB

קיבלנו אחוזים מעניינים בשיטת ההצפנה AES in mode of operation ECB כמוזכר לעיל , נרצה להבין אם יש מה לעשות בהקשר הזה :
למשל לבדוק מול datasets שונים , מול תמונות בגדלים שונים .
נרצה לשפר את אחוזי הדיוק שלו .

ב. PADDING

כמו שהוזכר בתיאור הכללי , עבדנו על לבדוק האם הלמידה של המודלים שמעבירים את הקלטים (התמונות) בפרמוטציה מצליחה רק בגלל שהתמונה קטנה מדי והמודל מצליח לזהות תבניות .
התחלנו לבדוק את זה ע"י הוספת ריפוד של פיקסלים לבנים סביב התמונות המקוריות ולתת למודל שמעביר בפרמוטציה ללמוד את התמונות האלו . (מודל A)
מצ"ב טבלת אחוזי השגיאה :

	image size	error rate
MNIST	28x28	3.70
	40x40	3.40
	60x60	3.30
Fashion-MNIST	28x28	12.30
	40x40	14.40
	60x60	10.80

כפי שניתן לראות אחוזי השגיאה יחסית נמוכים כלומר בגדלים שבדקנו עדיין המודל מצליח ללמוד . נרצה בהמשך לבדוק גדלים נוספים או לחשוב על דרכים אחרות לפזר את הפיקסלים בתמונה .

ג. ההצעה של Carlini

בחלק הקודם ציטטנו את המייל שניקולס קרליני שלח לנו ובמייל עצמו הוא מציע הצעה איך בכל זאת להצליח לתקוף (בשיטת הgrey-box), נרצה לנסות ולבדוק.

⇐ על המשך העבודה על הPadding , ועל סידור הקוד , ועל סיכום הפרוייקט השקענו בערך 40 שעות כל אחד .