Аналіз атак на лінійні методи машинного навчання

Середович В.В
Науковий керівник: Музичук Ю.А.

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА
Факультет прикладної математики та інформатики
Кафедра обчислювальної математики

29 травня 2020 р.

Постановка задачі
- Етапи роботи
- Модель

Методи атак
- FGSM
- I-FGSM
- DeepFool

Методи захисту
- Randomization
- Pixel Reflection

Аналіз алгоритмів
- Аналіз атак
- Аналіз захисту

Висновок

Постановка задачі

Змагальний приклад

Нехай існує класифікатор $f(x) : x \to y$, де $x \in X, y \in Y$, який передбачає значення y для вхідного x. Метою змагального прикладу є знайти такий x^*, який знаходиться в околі x, але хибно визначається класифікатором. Зазвичай максимальний рівень шуму в змагальному прикладі може бути не більше за певну L_p норму $\|x^* - x\|_p < \varepsilon$, де $p = 1, 2, \infty$. В межах даної роботи для визначення рівня пертурбації буде використовуватись саме L_∞ норма.

Етапи роботи

- Реалізувати лінійну модель машинного навчання
- Розглянути різні методи генерування змагальних прикладів
- Застосувати атаки на створену модель та проаналізувати їх ефективність
- Розглянути можливі методи захисту від атак

Модель

В ролі лінійного методу машинного навчання використовувався алгоритм мультикласової логістичної регресії.

Активаційна функція softmax

$$softmax(z_i) = \frac{e^{z_i}}{\sum_{k=1}^{C} e_k^z};$$

Вагова функція кросс-ентропії

$$\xi(Y, X) = \frac{1}{m} \sum_{i=1}^{m} \xi(y^{(i)}, x^{(i)}) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{C} y_j^{(i)} \log\left(\hat{y}_j^{(i)}\right) = J(\omega, b)$$

Для того щоб мінімізувати вагову функцію скористаємость алгоритмом градієнтного спуску.

FGSM

Ідея Fast Gradient Sign Method методу полягає в тому, щоб знайти такий змагальний приклад x^* який максимізує функцію втрати $J(x^*, y)$ до певного L_∞ обмеження. Цього можна досягти один раз використавши зворотне поширення:

$$X^* = X + \varepsilon \cdot sign(\Delta_x J(x, y)), \tag{1}$$

Для того щоб результат був в L_∞ ε - околі вхідного зображення, будемо використовувати функція образання: $Clip_{\boldsymbol{X},\varepsilon}\{\boldsymbol{X}^*\}$ - функція яка виконує по-піксельне обрізання зображення X^* так, $\boldsymbol{X}$.

$$Clip_{X,\varepsilon}\{\boldsymbol{X}^*\}(x, y, z) = min\Big\{255, \boldsymbol{X}(x, y, z) + \varepsilon, max\{0, \boldsymbol{X}(x, y, z) - \varepsilon, \boldsymbol{X}^*(x, y, z)\}\Big\} \tag{2}$$

де $\boldsymbol{X}(x, y, z)$ - це значення каналу z зображення $\boldsymbol{X}$ з координатами (x, y).

I-FGSM

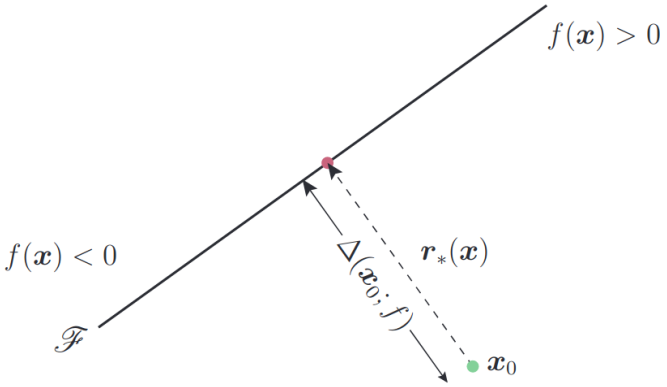
Ми застосовуємо звичайний метод декілька разів з певним невеликим кроком α і після кожного кроку використовуємо функцію обрізання, для того щоб переконатись що значення знаходяться в ε-околі оригінального зображення:

Algorithm 3 $I - FGSM$

Input: Приклад x, значення класу y_{true}, класифікатор f
Input: значення пертурбації ε, кількість ітерації T.
Output: Adversarial x^* з нормою $\|x^* - x\|_\infty \leq \varepsilon$;
$\alpha = \varepsilon / T$;
$x^* = x$;
for $t = 0$ *to* $T - 1$ **do**
    $x^* = Clip_{X,\varepsilon}\{x^* + \alpha \cdot sign(\Delta_x J(x, y))\}$;
end for
return $x^* = x^*_T$.

DeepFool

Для випадку бінарної класифікації легко бачити, що надійність моделі f в точці x_0, дорівнює відстані від x_0 до площини гіперпараметра $\mathscr{F} = \{x : \omega^T x + b = 0\}$, яка розділяє два класи.



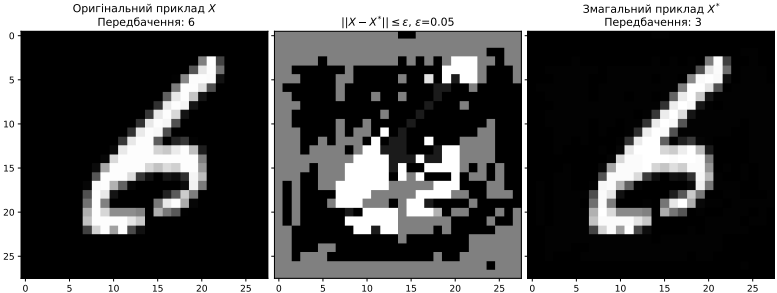


Рис.: Результат роботи алгоритму I-FGSM для $\varepsilon = 0.05$

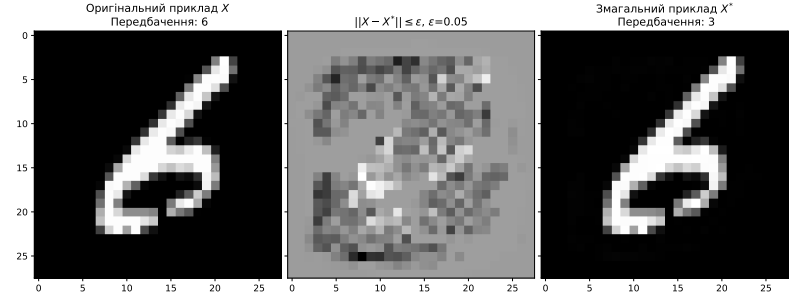
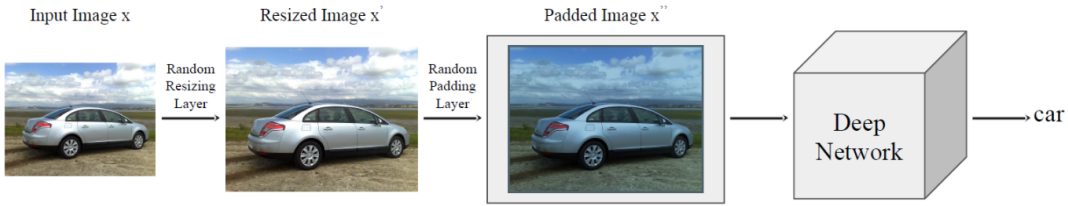


Рис.: Результат роботи алгоритму DeepFool для $\varepsilon = 0.05$

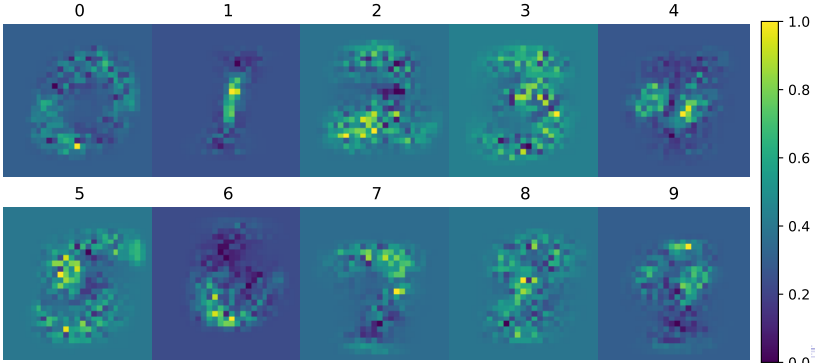
Randomization

1. Оригінальне зображення x розмірами $W \times H \times 1$ замінюють на нове x' з випадково вибраними розмірами $W' \times H' \times 1$.
2. Другим кроком буде випадкове наповнення деякого простору навколо зображення x' нулями, після чого буде утворене нове зображення x'' з розмірами $W'' \times H'' \times 1$.



Pixel Reflection

Ідея методу полягає в тому, щоб випадково вибрати піксель із зображення і потім, так само випадково замінити його на інший піксель в його малому околі. Для того щоб покращити цей алгоритм можна визначити найбільш важливі для класифікації класів місця зображення і уникати їх деформації, знижуючи ймовірність внесення туди змін. Для цього можна скористатись активаційною картою класів моделі. Для випадку нашої моделі вони будуть мати вигляд.



Аналіз атак

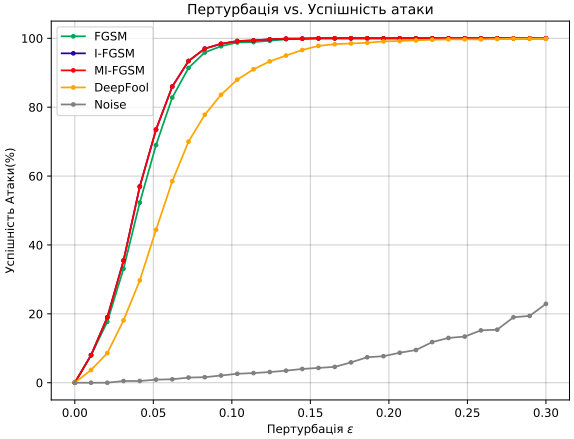


Рис.: Графік залежності успішності атаки від величини пертурбації. Нижньою межею буде виступати випадковий шум.

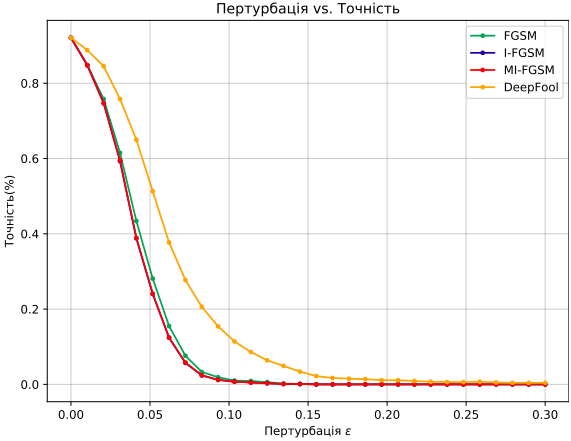


Рис.: Графік залежності успішності класифікації моделі від величини пертурбації. Початкова точність моделі $\sim 92\%$

Randomization

Алгоритм випадкової зміни розмірності опинився неефективним для захисту даної моделі. Навіть при дуже малих деформаціях зображень, класифікатор починав передбачати їх суттєво гірше і точність моделі знижувалась.

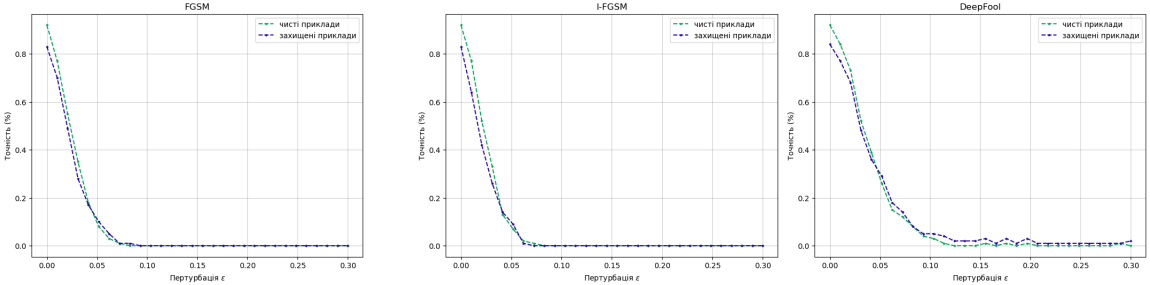


Рис.: Randomization

Pixel Deflection

Навіть при відносно великій кількості зсунутих пікселів модель майже не втрачала точність. При оптимальних параметрах деформації вхідного зображення, точність моделі для чистих зображень не падала зовсім, однак успішність проведених атак знижувалась досить слабо, а отже алгоритм не є надійним методом захисту.

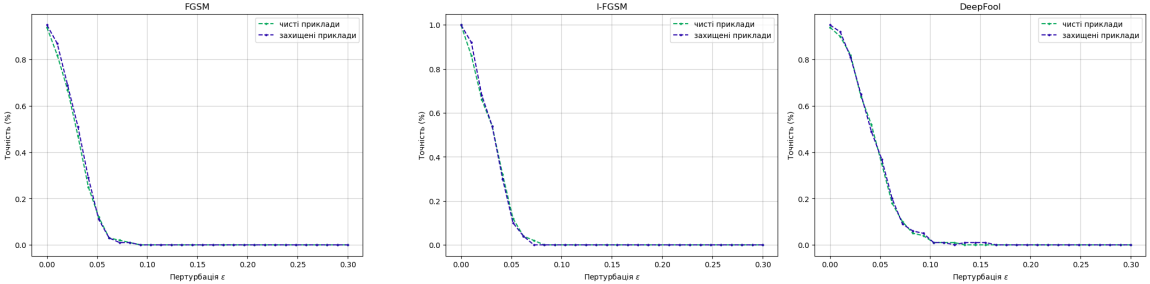


Рис.: Pixel Deflection

Висновок

В межах L_∞ норми найефективнішими методами опинились ітераційна модифікація та momentum оптимізація алгоритму FGSM. Також були розглянуті два методи захисту які базуються на ідеї руйнування структури змагальних прикладів через додавання в зображення контрольованого шуму. Методи захисту які базуються на ідеї випадкових деформацій зображень опинились не дуже ефективними у випадку лінійних моделей машинного навчання.