

Aspetti tecnici degli Adversarial Examples

I recenti progressi nel campo dell'Intelligenza Artificiale (IA) e del Machine Learning (ML) hanno portato a significativi avanzamenti in numerosi settori applicativi. Tuttavia, la ricerca ha evidenziato una classe di vulnerabilità intrinseca a questi sistemi: gli Adversarial Attacks.

Questi attacchi sfruttano le peculiarità dei modelli ML per generare input appositamente modificati, denominati Adversarial Examples, che inducono classificazioni errate pur rimanendo percettivamente indistinguibili dai dati originali per l'osservatore umano.

La presenza di Adversarial Examples solleva questioni critiche riguardo all'affidabilità e alla sicurezza dei sistemi IA/ML in applicazioni ad alto rischio.

Introduzione al Processo di Addestramento di un modello di Machine Learning

Il processo di addestramento di un modello di Machine Learning può essere descritto ad alto livello come composto da un modello ML di cui si è decisa l'architettura, di un set di parametri inizialmente ignoti e di un set di dati di addestramento. I parametri del modello ML sono inizialmente scelti con valori casuali e il processo di addestramento consiste nel calcolare il valore migliore di questi parametri (i modelli di Deep Learning hanno un elevato numero di parametri, ad esempio GPT-3/ChatGPT ne ha 175 miliardi).

Per far questo si esegue il modello ML sui dati di addestramento, di cui si conosce il risultato atteso, e si misura l'errore del risultato dell'esecuzione del modello ML rispetto al risultato corretto. Inizialmente questo errore è

molto grande, visto che il modello ML ha parametri casuali, ovvero non sa nulla. Poi si modificano i parametri in modo da ridurre l'errore.

Per semplicità si consideri un modello ML basato su Reti Neurali, addestrato con dati etichettati (Supervised Learning) e si indichi con $\{W\}$ il set dei parametri del modello e con $\{X\}$ il set dei dati di addestramento. L'errore del modello ML sui dati di addestramento è quindi una funzione di entrambi, $E(\{W\}, \{X\})$. Scopo dell'addestramento è cambiare il valore numerico dei parametri in modo da minimizzare l'errore, ovvero trovare $\min_{\{W\}} E(\{W\}, \{X\})$.

Metodo del Gradiente e Minimizzazione dell'Errore

Per trovare il migliore valore dei parametri, si utilizza principalmente il "metodo del gradiente" (Gradient Descent, Cauchy, 1847) o metodi simili. Questo è un metodo iterativo facilmente implementabile numericamente e quindi in software. Il metodo procede facendo piccolissime modifiche $\{\delta W\}$ ai parametri, $\{W\} \rightarrow \{W + \delta W\}$, tali che per ogni passo viene scelta la modifica che riduce maggiormente l'errore. Cauchy dimostrò che $\delta W = -\epsilon D\{W\}E(\{W\}, \{X\})$, ove ϵ è un numero piccolo (ad esempio 0,001) e $D\{W\}$ è la derivata della funzione errore rispetto ai parametri del modello ML. Il ciclo di calcolo (Epoca) su tutti i dati di addestramento viene ripetuto sino a quando non si raggiunge un errore molto piccolo, ad esempio inferiore all'1%.

Origine degli Adversarial Examples

Se è possibile addestrare il modello in modo che produca minimi errori, com'è possibile che vi siano Adversarial Examples? Bisogna ricordare che un modello ML apprende utilizzando uno specifico set di dati di addestramento che inizialmente non comprende gli Adversarial Examples. Il modello ML è capace di elaborare correttamente moltissimi dati che sono simili a quelli del set di addestramento, ma non

tutti.

Esistono particolari dati molto simili a quelli di addestramento sui quali il modello ML compie grandi errori: e proprio questi sono gli Adversarial Examples.

In semplici modelli di studio è possibile avere un'idea dell'origine degli Adversarial Examples [Rif. 8].

Costruzione degli Adversarial Examples

È possibile utilizzare un approccio simile a quello adottato per l'addestramento per cercare e costruire degli Adversarial Examples. Si assuma per semplicità di esposizione di aver calcolato i migliori parametri del modello ML e di non modificarli più. Invece si procede a modificare di poco i dati di addestramento cercando la modifica che genera il massimo errore, in formula $\max_{\delta X} E(\{W\}, \{X + \delta X\})$, ove δX deve rimanere molto piccolo.

Un semplice esempio di cosa si intende per “piccola modifica” può essere fatto per un modello ML che classifica immagini. Per il modello ML ogni immagine è una lunga lista di numeri, ogni numero corrisponde al colore di un pixel dell'immagine (questo se l'immagine è in toni di grigio, per immagini a colori ogni numero è il colore di un canale, ad esempio RGB, di un pixel).

Una “piccola modifica” può essere ad esempio modificare di poco il valore di un pixel e solo per 1 numero/pixel ogni 200 (cioè il 5 per mille); in molti casi questo tipo di modifica non è percettibile dall'occhio umano. In altre parole, si modificano alcuni pixel di un'immagine in modo che per l'occhio umano non vi sia alcuna differenza con l'immagine originale, ma in modo che il modello ML interpreti la nuova immagine come qualcosa di completamente diverso, ovvero massimo errore.

Sebbene il punto di partenza sia molto simile al processo di

addestramento, la matematica necessaria per risolvere questo problema è molto più complessa e una soluzione completa non è ancora nota.

Adversarial Examples: Esperimento con Convolutional Neural Network

Riportiamo i risultati di un esperimento di [Rif. 8] che utilizza un semplice modello Convolutional Neural Network per il riconoscimento di immagini e per il quale i ricercatori sono stati in grado di risolvere anche solo approssimativamente l'equazione che definisce gli Adversarial Examples. Il modello ML è addestrato su di un set di dati di addestramento (Train) e verificato su di un set di dati di prova (Test) su cui non è addestrato.

Per ogni ciclo di addestramento (Epoca) viene anche calcolato un set di Adversarial Examples (Adv), ovvero il set $\{X_{adv}=X+\delta X\}$ soluzione (approssimata) dell'equazione $\max_{\{\delta x\}} E(\{W\}, \{X+\delta X\})$ (il calcolo contestuale dei migliori valori dei parametri per minimizzare l'errore e delle variazioni dei dati di addestramento per massimizzare l'errore è chiamato un "problema min-max"). Il risultato è espresso nella seguente tabella (i numeri indicano la percentuale di errore espressa con un numero tra 0 e 1 per i tre set di dati Train, Test, Adv, per ogni ciclo/epoca di addestramento) [Rif. 8].



Tavola 1 [fonte Rif. 8]

Al termine dell'addestramento, l'errore sul set di dati Train è inferiore allo 0,2%, sul set di dati Test (su cui non viene fatto addestramento) è circa dell'1% e il programma ha costruito un set di dati di Adversarial Examples (Adv) molto simile al set di dati Train, ma su cui il modello ML sbaglia il riconoscimento dell'immagine il 71% delle volte.

Cosa è cambiato nelle immagini Adversarial Examples rispetto

alle immagini del set di Train? Spesso la modifica corrisponde all'aggiunta di "rumore bianco" all'immagine che può rendere un po' sfocati alcuni particolari e introdurre dei piccoli punti con colori errati in maniera apparentemente casuale (difetti dell'immagine o "glitch"), simile a quanto visto in Fig. 2. Nulla, comunque, che renda all'occhio umano la nuova immagine realmente diversa da quella originale.

Ri-addestramento con Adversarial Examples

Ora è possibile addestrare il modello ML anche utilizzando il set di dati di Adversarial Examples (Adv) che è stato generato, ma utilizzando sempre il set di dati Test solo per verificare l'addestramento. Il risultato è:



Tavola 2 [fonte Rif. 8]

Gli errori finali sui set di dati di Train e di Test sono cambiati ma le differenze non sono significative per l'esercizio svolto, mentre l'errore sul nuovo set di dati di Adversarial Examples (Adv) generato è sceso dal 71% a circa il 2,8%. Il modello ML ha quindi imparato a riconoscere correttamente anche gli Adversarial Examples generati con questo procedimento e che ora non sono più "antagonisti". Un modello ML che in pratica non ha Adversarial Examples è chiamato "robusto"; quindi questo modello ML può dirsi "robusto" almeno sino a quando non sarà trovata una nuova classe di suoi Adversarial Examples.

È importante sottolineare che dopo aver ri-addestrato il modello utilizzando anche il set di Adversarial Examples, si può generare con lo stesso procedimento un nuovo set di dati massimizzando l'errore, ma l'errore rimane molto piccolo e il modello riconosce correttamente anche questo set di dati. In altre parole, il procedimento adottato in questo esempio particolare per costruire un set di Adversarial Examples funziona solo la prima volta, poi non genera più set di Adversarial Examples.

Adversarial Examples: Limitazioni e Sfide Future

Riassumendo, è possibile minimizzare gli errori sui dati di addestramento calcolando il valore dei parametri $\{W\}$ e al contempo generare un set di dati $\{X+\delta X\}$ quasi indistinguibile dal set di dati di addestramento ma che massimizza l'errore del modello ML. Anche se poi è possibile ri-eseguire il processo di addestramento includendo gli Adversarial Examples individuati creando un modello ML robusto, ad oggi non è noto un metodo per generare tutti gli Adversarial Examples di un modello ML, ovvero garantire che gli Adversarial Examples trovati siano tutti quelli di un modello ML.

Inoltre, come indicato precedentemente, aggiornare l'addestramento di un modello ML può portare a nuovi problemi, ad esempio la "smemoratezza" (Forgetfulness, dimenticare informazioni precedentemente acquisite) o una maggiore predisposizione ad attacchi di inversione e estrazione di dati di addestramento, problematiche rilevate in alcuni modelli ML di studio.

Il messaggio che viene da questo esercizio è che il processo di addestramento attuale non è completo: l'esistenza degli Adversarial Examples non è dovuta a qualche disattenzione o peculiarità di alcuni modelli ML particolari, ma è una caratteristica di come sono costruiti i modelli ML attuali. La ricerca scientifica su questo problema è molto intensa [Rif. 1], sono stati introdotti anche standard e benchmark sulla "robustezza" dei modelli ML e sui metodi per renderli robusti [Rif. 9], ma è probabile che per poter trovare una soluzione definitiva sia necessaria una migliore comprensione (soprattutto matematica) dei modelli di Machine Learning.

Riferimenti Bibliografici

Rif. 1: N. Carlini, "A Complete List of All (arXiv)

Adversarial Example Papers", <https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html> , nel solo anno 2022 risultano pubblicati quasi 3.000 articoli scientifici relativi a "Adversarial Machine Learning"

Rif. 8: Z. Kolter, A. Madry, "Adversarial Robustness – Theory and Practice", <https://adversarial-ml-tutorial.org/>

Rif. 9: F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, M. Hein, "RobustBench: a standardized adversarial robustness benchmark", <https://robustbench.github.io/> , arXiv:2010.09670



Continua a leggere: scarica il white paper gratuito **"Adversarial Attacks a Modelli di Machine Learning"**

Articolo a cura di **Andrea Pasquinucci**

Profilo Autore



PhD CISA CISSP

Consulente freelance in sicurezza informatica: si occupa prevalentemente di consulenza al top management in Cyber Security e di progetti, governance, risk management, compliance, audit e formazione in sicurezza IT.

Condividi sui Social Network:

<https://www.ictsecuritymagazine.com/articoli/gli-adversarial-examples/>