Ataques adversarios

E. David Guzmán Ramírez

Licenciatura en Ciencia de Datos
Introducción al Aprendizaje Profundo

M. en C. Berenice Montalvo Lezama
M. en C. Ricardo Montalvo Lezama

14 de junio de 2021

Contenidos

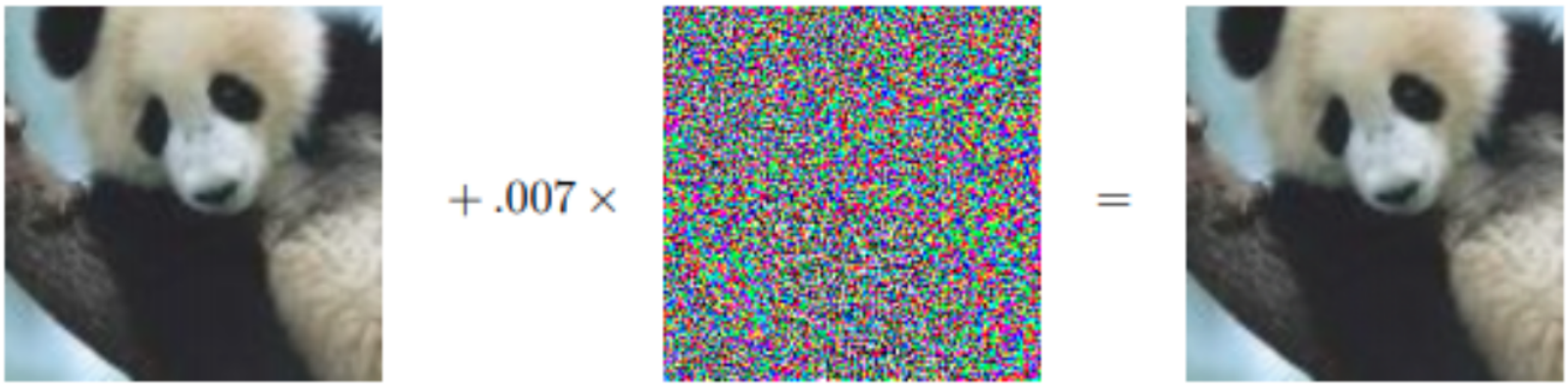
Introducción

Las redes neuronales profundas son modelos poderosos que se han utilizado ampliamente para lograr un rendimiento cercano al nivel humano en una variedad de tareas.

Sin embargo, a pesar de su desempeño superior estudios recientes han encontrado que incluso los modelos del estado del arte son sumamente vulnerables a ataques adversarios.

Introducción

Un ataque adversario es una muestra de datos de entrada que ha sido perturbada levemente con la intención de hacer fallar a un clasificador.



Panda Ruido Gibón

Figura 1: Ejemplo de ataque adversario. Imagen tomada de *Explaining and harnessing adversarial examples.*

Goodfellow et al., *Explaining and harnessing adversarial examples*, 2016.

Introducción

Figura 2: Ejemplo de ataque adversario en el que es posible engañar a la red cambiando únicamente un pixel. Imagen tomada de *One Pixel Attack for Fooling Deep Neural Networks*.

Su et al., *One Pixel Attack for Fooling Deep Neural Networks*, 2019.

Introducción

Los escenarios de posibles ataques adversarios se pueden clasificar en diferentes maneras:

- **Ataque no dirigido:** el objetivo es hacer que el clasificador prediga una etiqueta incorrecta, la etiqueta incorrecta específica no importa.
- **Ataque dirigido:** el objetivo es cambiar la predicción del clasificador a alguna clase objetivo específica.

Introducción

En segundo lugar, los escenarios de ataque se pueden clasificar por la cantidad de conocimiento que el adversario tiene sobre el modelo:

- **Caja negra:** el atacante no sabe mucho sobre el modelo, pero puede sondear o consultar el modelo, es decir, darle algunas entradas y observar salidas.
- **Caja blanca:** el atacante tiene pleno conocimiento del modelo, como la arquitectura del modelo y los valores de todos los parámetros y pesos entrenables.

Motivación

Los ataques adversarios plantean problemas de seguridad porque podrían usarse para realizar un ataque a los sistemas de aprendizaje profundo, incluso si el atacante no tiene acceso al modelo subyacente.

Con la introducción de modelos de aprendizaje profundo en cada vez más distintos aspectos de nuestra vida, los problemas que estos ataques adversarios pueden ocasionar son preocupantes.

Motivación

Figura 3: Ataque adversario al sistema de navegación autónomo de un Tesla, en el que confunde un señalamiento de velocidad de 35 millas/hora por 85 millas/hora. Figura tomada de MIT Technology Review: Trick a Tesla into accelerating by 50 miles per hour.

Descripción del problema

Para acelerar la investigación sobre ataques de adversarios, Google Brain organizó la *Competencia de Ataques y Defensas Adversarios* en la edición de 2017 de NIPS, la cual está disponible en Kaggle, la competencia a su vez constaba de tres subcompeticiones:

- Ataques adversarios no dirigido.
- Ataques adversarios dirigido.
- Defensas contra ataques adversarios.

Por el momento me concentré en los ataques adversarios no dirigidos, la cual trata de un ataque de caja negra no dirigido, es decir, dada una imagen de entrada generar una imagen adversaria que engañe a un clasificador desconocido.

Análisis exploratorio

El dataset para esta competencia debía cumplir con 3 aspectos:

1. Conjunto de datos suficientemente grande y problema no trivial.
2. Problema bien conocido, por lo que las personas potencialmente pueden reutilizar los clasificadores existentes.
3. Muestras de datos que nunca se usaron antes.

El conjunto de ImageNet cumple con los primeros dos requisitos, posteriormente se etiquetaron 1000 nuevas imágenes compatibles con ImageNet las cuales servían como el conjunto de datos de desarrollo para la competencia y para cumplir el tercer requisito.

Desafortunadamente, nunca hicieron público el dataset de evaluación ni el modelo de caja negra que había que engañar, por lo que no hay manera de comparar mis resultados con los de la competencia.

Propuesta de solución

La idea es tratar de implementar varias técnicas de ataques adversarios[1] sobre las arquitecturas existentes, particularmente usaré los siguientes cuatro métodos:

- **FGSM (Fast Gradient Sign Method):** la idea es generar el ejemplo adversario $\boldsymbol{x}_{\text{adv}}$ de la siguiente forma

$$\boldsymbol{x}_{\text{adv}} = \boldsymbol{x} + \epsilon \cdot \text{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right).$$

  donde ϵ es el orden de la perturbación, J es la función de pérdida (usualmente cross-entropy), $\boldsymbol{\theta}$ los pesos del modelo, $\boldsymbol{x}$ la imagen original y y la etiqueta.

- **PGD (Projected Gradient Descent):** podemos verlo como una variante de varios pasos, donde α es la magnitud de la perturbación en cada paso

$$\boldsymbol{x}_{\text{adv}}^{t+1} = \text{Proj}\left(\boldsymbol{x}_{\text{adv}}^{t} + \alpha \cdot \text{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right)\right)$$

[1] Kui Ren et al., *Adversarial Attacks and Defenses in Deep Learning*, 2020.

Propuesta de solución

- **MIFGSM (Momentum Iterative Fast Gradient Sign Method):** algoritmo ganador de esta competencia, inspirados por los optimizadores con momento proponen

$$\boldsymbol{x}_{\text{adv}}^{t+1} = \text{Clip}\left(\boldsymbol{x}_{\text{adv}}^{t} + \alpha \cdot \text{sign}(g_{t+1})\right),$$

donde $g_{t+1} = \xi \cdot g_t + \dfrac{\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)}{\|J(\boldsymbol{\theta}, \boldsymbol{x}, y)\|}$, donde ξ es un factor de decaimiento.

- **One Pixel Attack:** plantea el problema como un problema de optimización

$$\max_{\boldsymbol{x}_{\text{adv}}} f_{\text{adv}}(\boldsymbol{x} + \boldsymbol{x}_{\text{adv}}),$$

$$\text{con la restricción } \|\boldsymbol{x}_{\text{adv}}\| \leq d,$$

donde $d = 1$ para el caso de un ataque a un pixel. Particularmente proponen resolver este problema de optimización con *evolución diferencial*.

Propuesta de solución

Hay una variedad de arquitecturas preentrenadas en PyTorch con el conjunto de datos de ImageNet que sirven a la perfección para esta tarea, particularmente usaré las siguientes

- AlexNet
- Resnet18
- Inception v3
- MobileNet v2

Con estas arquitecturas y métodos se pueden explorar los ataques y defensas adversarias.

Resultados

Con los ataques de caja blanca, donde los ejemplos adversarios se hacen a la medida de cada modelo, la disminución en el accuracy es enorme. Particularmente el ataque a un sólo pixel no es tan eficiente, pero en cualquier caso logra bajar el accuracy.

Modelo	**Limpio (acc@1/acc@5)**	**FGSM (acc@1/acc@5)**	**PGD (acc@1/acc@5)**	**MIFGSM (acc@1/acc@5)**	**OnePixel (acc@1/acc@5)**
AlexNet	60.9 / 84.6	6.0 / 28.6	2.9 / 19.3	3.5 / 21.0	58.3 / 83.4
ResNet-18	82.5 / 95.4	3.5 / 24.3	0.8 / 14.6	1.0 / 13.5	78.7 / 94.5
Inception v3	76.5 / 93.1	10.8 / 39.3	3.9 / 28.1	5.1 / 27.6	70.1 / 91.8
MobileNet v2	85.0 / 97.3	3.5 / 24.4	0.5 / 11.3	0.6 / 8.8	81.4 / 96.6

Tabla 1: Ataques de caja blanca.

Resultados

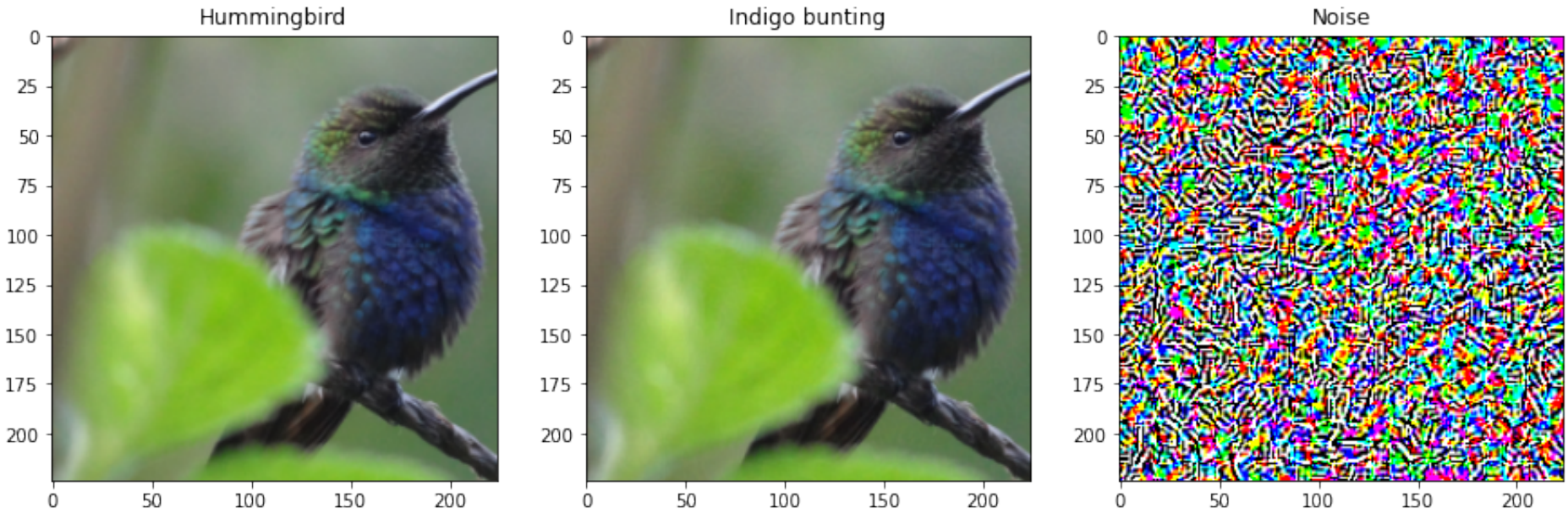


Figura 4: Ataque a AlexNet con FGSM.

Resultados

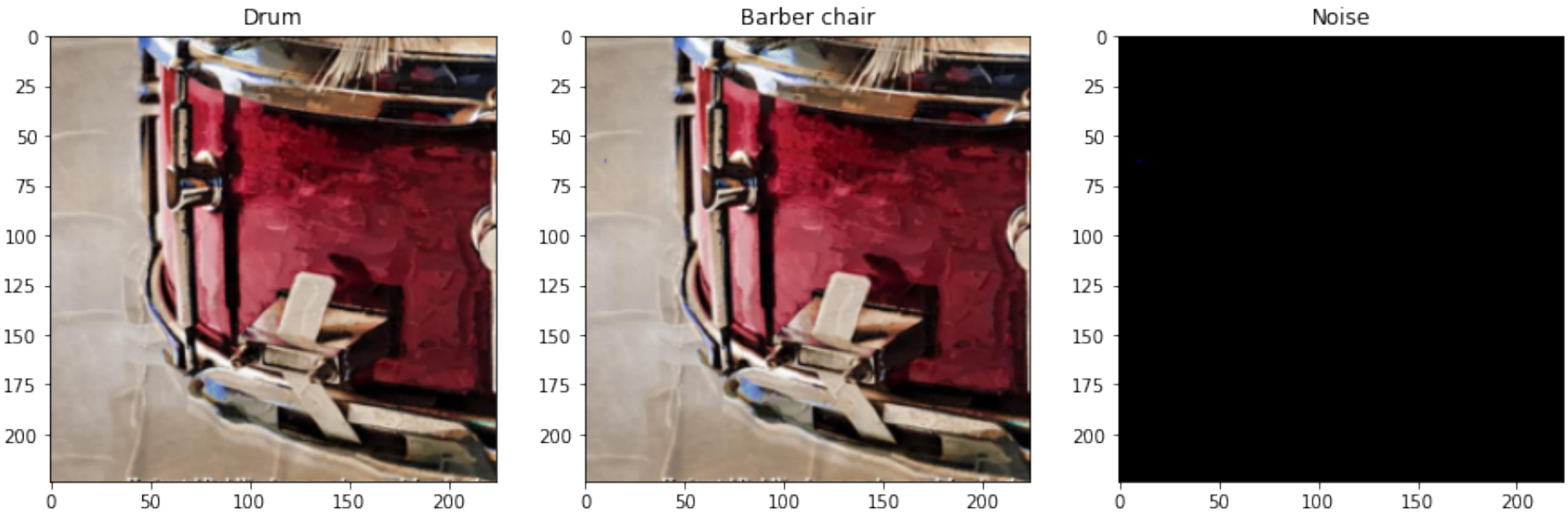


Figura 5: Ataque a Inception v3 con OnePixel.

Resultados

Para el ataque de caja negra generé los ejemplos adversarios con Inception v3 y los probé en MobileNet v2, el cual funcionaba como mi modelo de caja negra. En este caso fui más agresivo con los ataques. Aunque no es tan efectivo como un ataque de caja blanca, se logra bajar considerablemente el accuracy del modelo.

Modelo	**Limpio (acc@1/acc@5)**	**FGSM (acc@1/acc@5)**	**PGD (acc@1/acc@5)**	**MIFGSM (acc@1/acc@5)**	**OnePixel (acc@1/acc@5)**
MobileNet v2	85.0 / 97.3	51.7 / 78.3	59.4 / 82.9	56.3 / 81.5	83.5 / 97.0

Tabla 2: Ataque de caja negra a MobileNet v2.

Resultados

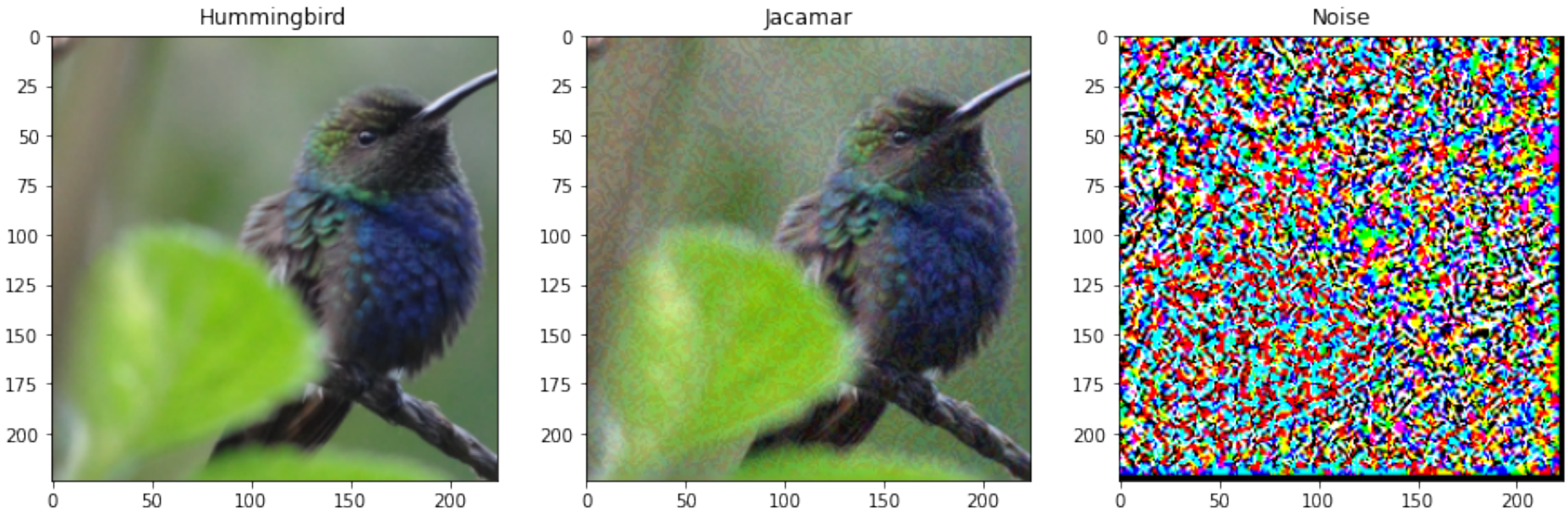


Figura 6: Ataque de caja negra a MobileNet v2 con FGSM.

Resultados

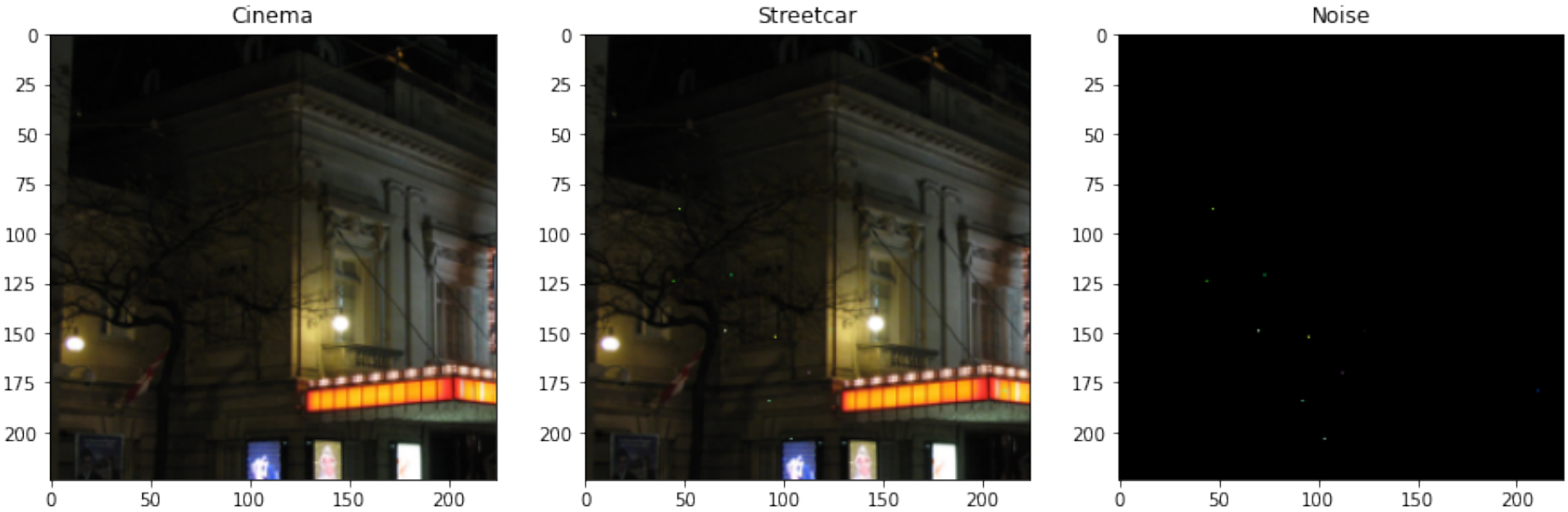


Figura 7: Ataque de caja negra a MobileNet v2 con OnePixel.

Resultados

Una de las formas de hacer más robustos los modelos a ataques adversarios es agregar ejemplos adversarios al conjunto de datos de entrenamiento. Para probar, agregué ejemplos adversarios al conjunto de CIFAR10 y entrené una versión modificada de MobileNet v2 sin y con los ejemplos adversarios.

Modelo	**Limpio (accuracy)**	**FGSM (accuracy)**	**PGD (accuracy)**	**MIFGSM (accuracy)**	**OnePixel (accuracy)**
MobileNet v2	81.68	37.03	28.29	31.27	63.17
MobileNet v2 adversarial	80.62	77.44	77.49	77.35	77.58

Tabla 3: Entrenamiento sin y con ejemplos adversarios en CIFAR10 con una versión modificada de MobileNet v2.

Resultados

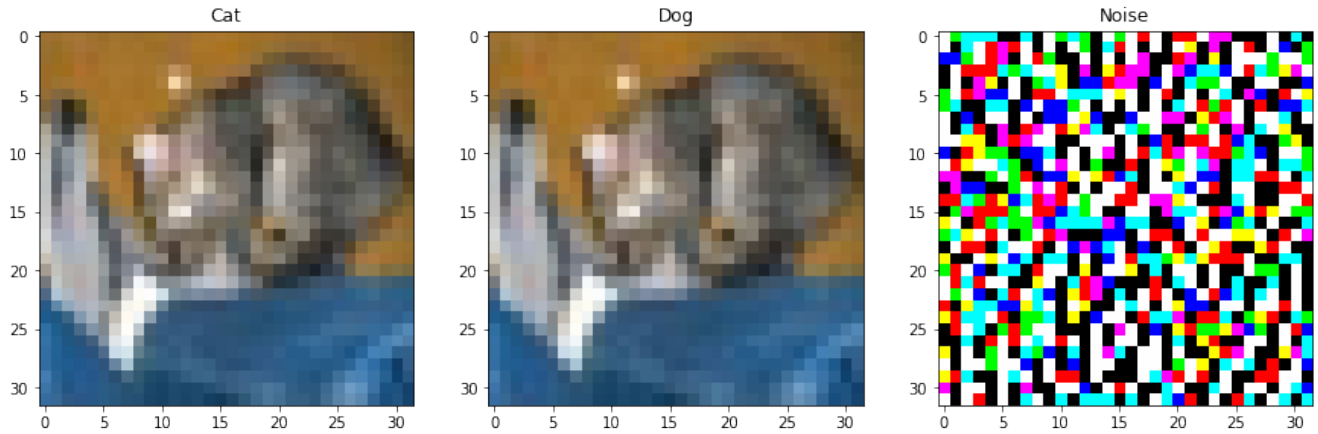


Figura 8: Ataque adversario con FGSM a MobileNet v2 entrenada sin ejemplos adversarios.

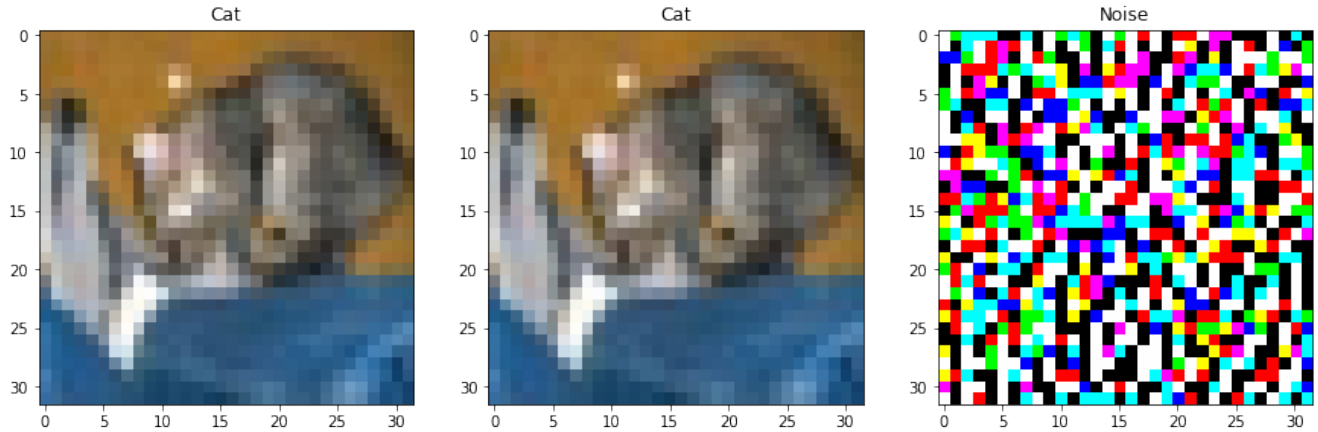


Figura 9: Ataque adversario con FGSM a MobileNet v2 entrenada con ejemplos adversarios.

Conclusiones

- Los ataques adversarios son un fenómeno interesante y un problema importante en la seguridad del aprendizaje automático, por lo que es relevante hacer notar a la comunidad de este problema.
- Ataques relativamente simples pueden engañar fácilmente incluso a los modelos más recientes sin que un humano llegue a notarlos.
- El estudio de estos ataques a su vez nos puede ayudar a generar defensas para hacer a los modelos más robustos, confiables y seguros.