



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO  
MAESTRÍA EN CIENCIAS MATEMÁTICAS

---

# Interpretabilidad de los Ataques Adversarios

---

REPORTE DEL PROYECTO FINAL  
- REDES NEURONALES -

Aaron Kelley

Rodrigo Fritz

18 de junio, 2021

# Abstract

Abstract Goes here

Keywords:

## Índice

<b>1. Introducción</b>	<b>2</b>
<b>2. Método</b>	<b>3</b>
2.1. Datos . . . . .	3
2.1.1. MNIST . . . . .	3
2.1.2. CIFAR-10 . . . . .	4
2.2. Ataques . . . . .	6
2.2.1. Fast Gradient Method . . . . .	6
2.2.2. Carlini & Wagner . . . . .	6
2.3. Defensas . . . . .	6
2.3.1. Compresión JPEG . . . . .	6
<b>3. Resultados</b>	<b>8</b>
3.1. Las Redes como un Filtro . . . . .	8
3.2. ¿Que Hace la Defensa JPEG? . . . . .	8
3.3. Los Efectos de Overfitting y Overparameterization . . . . .	8
3.4. Saliency . . . . .	9

## Development Notes/Ideas

- maybe we should take out Projected Gradient Descent
- with cleverhans we can do targeted attacks with Fast Gradient Descent

# 1. Introducción

Las redes neuronales profundas (DNNs) han ganado una alta reputación con respecto a sus capacidades de clasificar imágenes igual (o hasta superior) que los humanos. Pero en el pasado reciente, ha quedado cada vez más claro que las redes aprenden a clasificar de manera muy distinta que los humanos. Una de las propiedades que realmente demostró eso fue el descubrimiento de su susceptibilidad a los ataques adversarios [13]. Esos ataques se construyen agregándoles a las imágenes una pequeña perturbación imperceptible para los humanos que engaña a la red. Es decir, aunque una imagen adversaria para nosotros parezca igual a la original, la red se equivoca con la clasificación de la adversaria con alta probabilidad. Como los ataques pueden no ser detectables por los humanos, se plantea la preocupación que puedan ser usados maliciosamente; por ejemplo, en la tecnología de reconocimiento de imágenes que se utiliza en los automóviles autónomos. Por eso se requiere más profundización del conocimiento asociado.

A grandes rasgos, los ataques pueden dividirse entre dos categorías: dirigidos y no-dirigidos. Primero hablemos de los ataques dirigidos; muchos usan el gradiente de manera directa. La idea es que se toma el gradiente de la función de pérdida con respecto a la imagen, y eso se usa para diseñar una perturbación que maximiza el error de la clasificación cuando se le agrega a la imagen. Dos ejemplos de ataques no-dirigidos que utilizan el gradiente directamente son el projected gradient descent (PGD) [9] y el fast gradient sign method (FGSM) [3]. Este último saca el signo de cada elemento del gradiente en lugar de usar el gradiente verdadero; eso hace que sea más rápido con grandes cantidades de datos. En este artículo se usan tanto PGD como FGSM para explorar los ataques no-dirigidos. El objetivo de los ataques no dirigidos es que cambien la clasificación correcta a cualquier otro ataque, mientras que los ataques dirigidos tienen una clasificación deseada a la que cambian la clasificación correcta. Se examina el ataque Carlini & Wager (CW), el cual puede actuar como dirigido o no-dirigido.

Cabe mencionar la diferencia entre los ataques de caja blanca y caja negra. En el primero, todos los detalles de la red (pesos, arquitectura, etc) son conocidos por el atacante. Por el contrario, en los ataques de caja blanca, los detalles son escondidos, y solo se conocen las entradas (inputs) y las salidas (outputs). Aunque parezca muy difícil, los ataques de caja negra son bastante exitosos por la facilidad de aproximar el gradiente solo por las entradas y las salidas. Los ataques que se emplean en este artículo son los de caja blanca.

Desde el descubrimiento de esa vulnerabilidad que tienen las DNN, se han diseñado defensas para tratar de combatir los ataques. Aunque ninguna defensa funciona para resistir completamente los ataques, muchas sí tienen efecto, y vale la pena explorar qué propiedades contribuyen a su éxito. Al igual que los ataques, las defensas también se pueden categorizar en dos modalidades. Las del primer tipo modifican el entrenamiento de la red para que la función de pérdida se vuelva más suave. Entre más suave esa función, más difícil será para que las perturbaciones pequeñas hagan cambios grandes. Un ejemplo conocido de este tipo es el entrenamiento adversario [3, 11, 13]. Las defensas del segundo tipo, y la que se estudia en este artículo, no modifican el entrenamiento ni la arquitectura, sino que utilizan un preprocesamiento en las imágenes de entrada. La defensa que se utiliza en este artículo es compresión JPEG [2].

## 2. Método

### 2.1. Datos

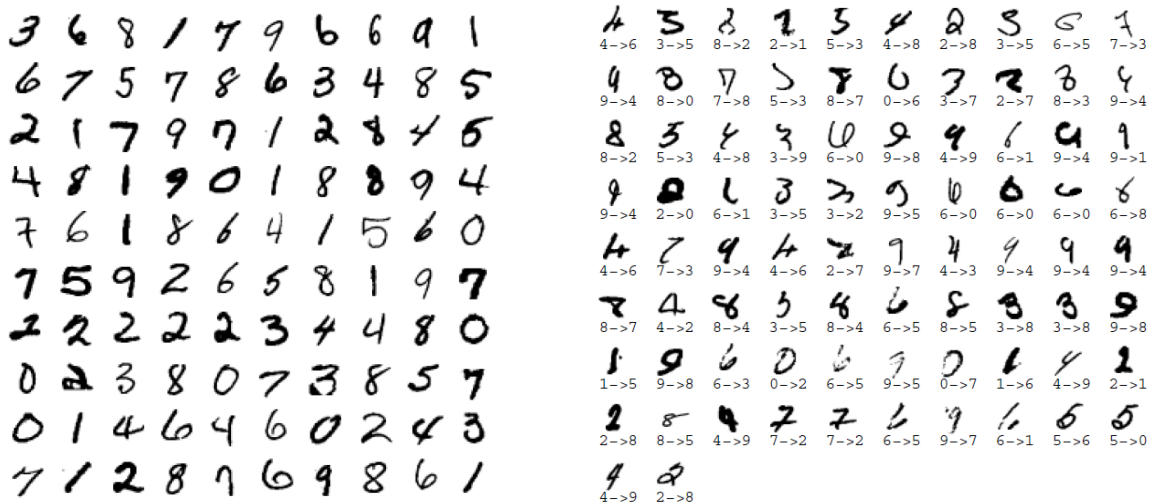
Para entrenar nuestras redes y probar los ataques adversarios y las defensas contra ellos, se emplearon las dos bases de datos más sencillas para procesamiento de imágenes: MNIST y CIFAR-10.

#### 2.1.1. MNIST

La base de datos MNIST es un compendio de los dígitos del 0 al 9 en letra manuscrita con 60,000 imágenes para entrenar a distintos sistemas de procesamiento de imágenes y 10,000 imágenes para evaluarlos. Se trata de un subconjunto de imágenes de un conjunto más grande que compiló el National Institute of Standards and Technology (NIST) del Departamento de Comercio los EEUU. Las siglas MNIST database significan Modified NIST database. Es una buena base de datos para probar técnicas de aprendizaje y métodos de reconocimiento de patrones con datos del mundo real empleando un esfuerzo mínimo en preprocesamiento y formato.

De las 10,000 imágenes de evaluación, la mitad fueron escritas por estudiantes de preparatoria y la otra mitad por empleados de la oficina de censos, mientras que de las 60,000 imágenes de entrenamiento, 58,527 de los números fueron escritos por 500 estudiantes y el resto por los empleados.

El tamaño de estas imágenes en blanco y negro fue normalizado a una caja de  $20 \times 20$  píxeles y se colocó su centro de masa en un campo de  $28 \times 28$  [6]. En la Figura 1 se muestran algunos ejemplos de estas imágenes.

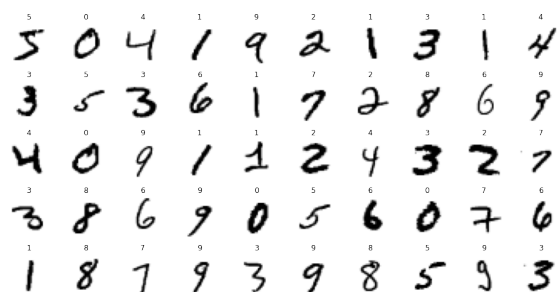


(a) Algunas imágenes del conjunto de evaluación de la base de datos MNIST [7].

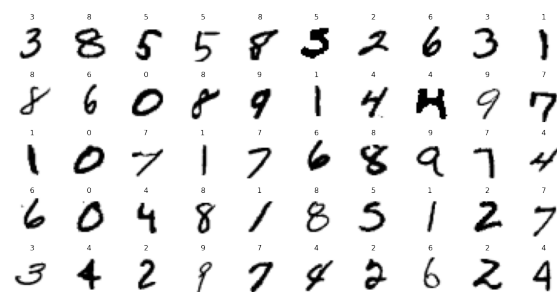
(b) 82 imágenes de evaluación que LeNet-5 clasificó erróneamente [7].

Figura 1: Imágenes de MNIST.

La distribución de los dígitos en MNIST no es homogénea, como puede observarse en la Figura 4.

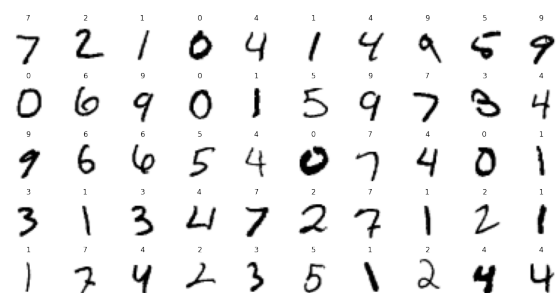


(a) Primeras 50 imágenes de MNIST con sus respectivas etiquetas.

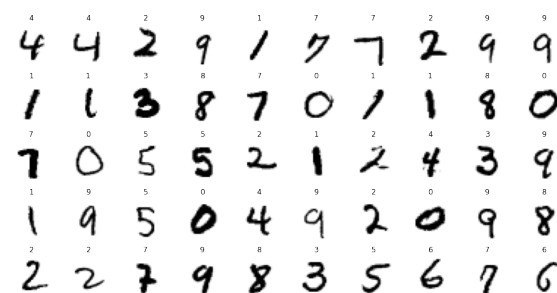


(b) 50 imágenes aleatorias de MNIST con sus respectivas etiquetas.

Figura 2: Imágenes del conjunto de entrenamiento (train) de MNIST.

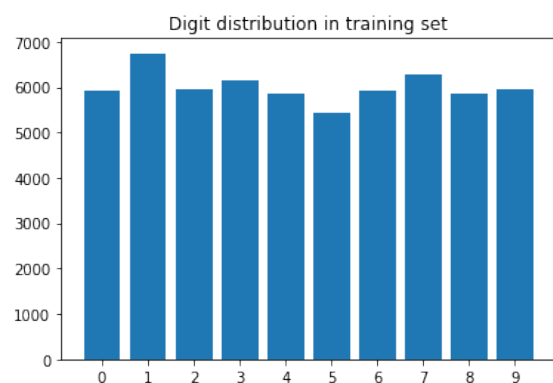


(a) Primeras 50 imágenes de MNIST con sus respectivas etiquetas.

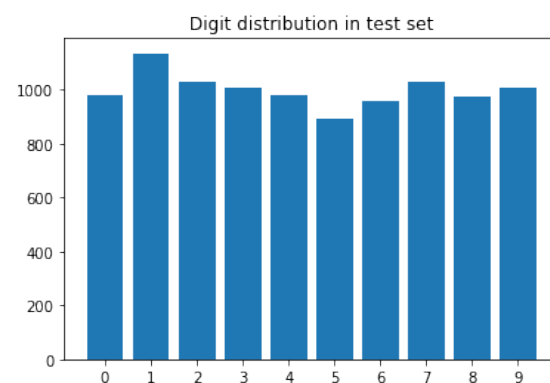


(b) 50 imágenes aleatorias de MNIST con sus respectivas etiquetas.

Figura 3: Imágenes del conjunto de evaluación (test) de MNIST.



(a) Distribución de los dígitos en el conjunto de entrenamiento de MNIST.



(b) Distribución de los dígitos en el conjunto de evaluación de MNIST.

Figura 4: Distribución de los dígitos en MNIST.

## 2.1.2. CIFAR-10

Los grupos del MIT y la NYU recopilaron un conjunto de millones de diminutas imágenes en color de la web, se trata de un excelente conjunto de datos para el entrenamiento no supervisado de modelos generativos profundos. Se crearon dos juegos de etiquetas confiables: el conjunto CIFAR-10, que tiene 6000 ejemplos de cada una de 10 clases y el conjunto CIFAR-100, que tiene 600 ejemplos de cada una de 100 clases que no se

superponen. Usando estas etiquetas, se mostró que el reconocimiento de objetos mejora significativamente al entrenar previamente una capa de características en un gran conjunto de imágenes diminutas sin etiquetar.

Esta base de datos fue ensamblada buscando en la web imágenes de cada sustantivo en inglés no abstracto en la base de datos léxica WordNet. Utilizaron varios motores de búsqueda, incluidos Google, Flickr y Altavista y mantuvieron aproximadamente los primeros 3000 resultados para cada término de búsqueda. Después de recopilar todas las imágenes para un término de búsqueda en particular, eliminaron duplicados perfectos e imágenes en las que una parte excesivamente grande de los píxeles eran blancos, ya que tendían a ser figuras sintéticas en lugar de imágenes naturales. El término de búsqueda utilizado para encontrar una imagen le proporciona una etiqueta aproximada, aunque es extremadamente poco confiable debido a la naturaleza de la tecnología de búsqueda de imágenes en línea. En total, el conjunto de datos contiene 80 millones de imágenes en color reducidas a  $32 \times 32$  y distribuidas en 79,000 términos de búsqueda [4].

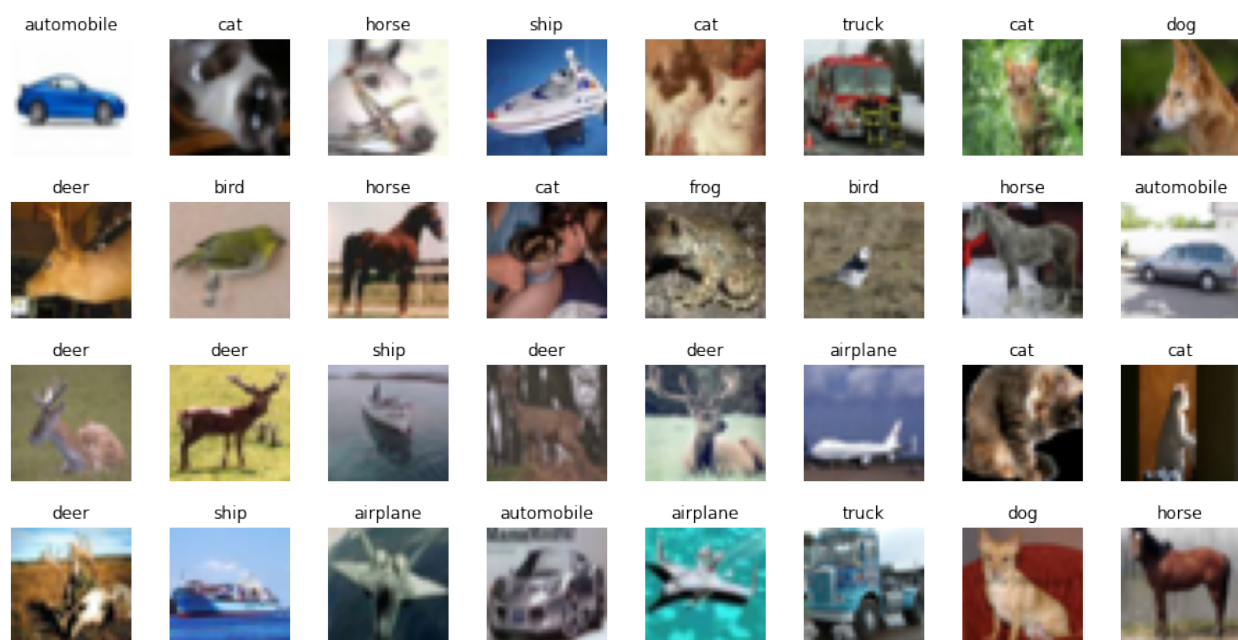


Figura 5: Algunas imágenes del conjunto de CIFAR-10 [4]

El conjunto de datos se divide en cinco lotes de entrenamiento y un lote de prueba, cada uno con 10,000 imágenes. El lote de prueba contiene exactamente 1000 imágenes seleccionadas al azar de cada clase. Los lotes de entrenamiento contienen las imágenes restantes en orden aleatorio, pero algunos lotes de entrenamiento pueden contener más imágenes de una clase que de otra. Entre ellos, los lotes de entrenamiento contienen exactamente 5000 imágenes de cada clase [5].

Las 10 clases de CIFAR-10 se encuentran en orden alfabético y son:

- |               |         |         |          |           |
|---------------|---------|---------|----------|-----------|
| 1. airplane   | 3. bird | 5. deer | 7. frog  | 9. ship   |
| 2. automobile | 4. cat  | 6. dog  | 8. horse | 10. truck |

## 2.2. Ataques

### 2.2.1. Fast Gradient Method

[maybe more, 3]

Sean  $\theta$  los parámetros de un modelo,  $x$  la entrada,  $y$  las salidas asociadas, y  $J(\theta, x, y)$  la función de costo. La función de costo se linealiza alrededor del valor actual de  $\theta$ . Sea  $\epsilon \in \mathbb{R}^+$ . Definamos la imagen adversaria

$$\tilde{x} = x + \epsilon \eta_{\text{opt}}$$

Se puede definir  $\eta_{\text{opt}}$  por el problema de optimización

$$\eta_{\text{opt}} = \underset{\eta}{\operatorname{argmax}} \left\{ \operatorname{grad}^\top \eta : \|\eta\|_p < \epsilon \right\}$$

Donde  $p \in \mathbb{N} \cup \{\infty\}$  y  $\operatorname{grad} = \nabla_x J(\theta, x, y)$ . Experimentamos con tres valores de  $p$ :

a)  $p = 1$ , no lo sé, pero se encuentra en el código

b)  $p = 2$ ,

$$\eta_{\text{opt}} = \frac{\operatorname{grad}}{\|\operatorname{grad}\|}$$

c)  $p = \infty$ ,

$$\eta_{\text{opt}} = \operatorname{sign}(\nabla_x J(\theta, x, y))$$

### 2.2.2. Carlini & Wagner

[1]

## 2.3. Defensas

### 2.3.1. Compresión JPEG



Figura 6: Calidad JPEG

JPEG es una compresión con pérdida que utiliza la transformada discreta del coseno (DCT) y normalmente elimina muchos componentes de alta frecuencia, a los que la percepción humana es menos sensible [2, 12]. Consta de los siguientes pasos:

1. Conversión de la imagen de formato RGB a formato  $YC_bC_r$ , donde el canal Y representa luminancia y los canales  $C_b$  y  $C_r$  representan crominancia. Esto está hecho porque el sistema visual humano se basa más en el contenido espacial y la agudeza que en el color para la interpretación.
2. Submuestreo espacial de los canales de crominancia en el espacio  $YC_bC_r$ : el ojo humano es mucho más sensible a los cambios de luminancia, y reducir la muestra de la información de crominancia no afecta mucho la percepción del ser humano de la imagen.
3. Dividir la imagen en bloques de  $8 \times 8$  y aplicar DCT en 2D a cada bloque. Esto se hace para cada canal por separado. Este paso produce mayor compresión de los datos de la imagen.
4. Cuantificación de las amplitudes de frecuencia, lograda dividiendo cada término de frecuencia por una constante (diferente) y redondeándola al número entero más cercano. Como resultado, muchos componentes de alta frecuencia generalmente se establecen en cero y otros se reducen. La cantidad de compresión se rige por un parámetro de calidad especificado por el usuario, que define la reducción en la resolución. Aquí es donde el algoritmo JPEG logra la mayor parte de la compresión, a expensas de la calidad de la imagen. Este paso suprime más las frecuencias más altas, ya que estos coeficientes contribuyen menos a la percepción humana de la imagen.
5. Compresión sin pérdidas de los datos del bloque.



### 3. Resultados

#### 3.1. Las Redes como un Filtro

- Bode Diagrams

#### 3.2. ¿Que Hace la Defensa JPEG?

- Intuition: JPEG removes noise that humans can't see
- Show graphs of jpeg defense vs epsilon of each attack

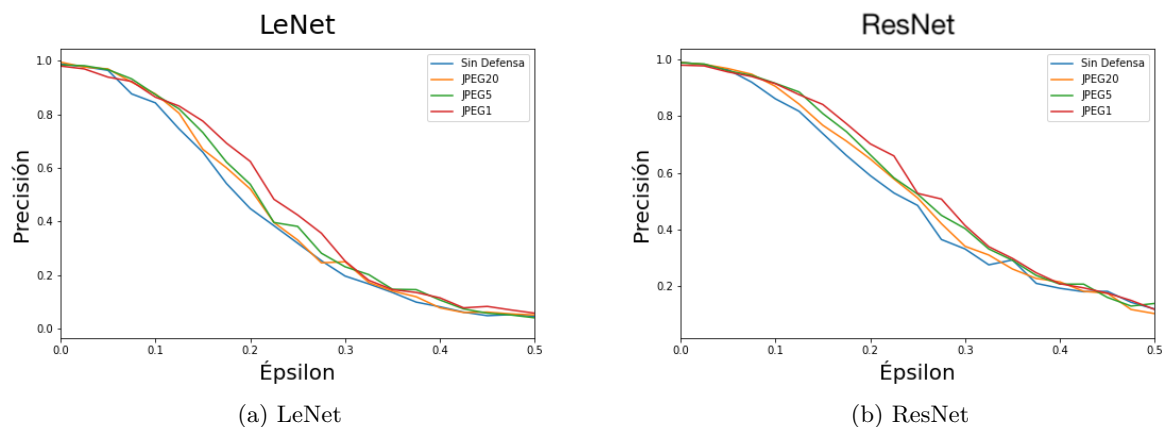


Figura 7: jpeg defense

- bode diagrams with jpeg conversion as first layer

#### 3.3. Los Efectos de Overfitting y Overparameterization

- here they suggest overparameterization contributes to sharp gradient landscape [8]...we find the opposite

En la Figura 8a se observa que conforme aumentamos el número de épocas (epochs), la precisión (accuracy) aumenta con respecto a cada epsilon, excepto para 200 épocas, cuya precisión es menor que la de 100 épocas, e incluso menor que la de 50 épocas para epsilon > 0.5. Aumentar el número de épocas implica un sobreajuste (overfitting) de la red neuronal, lo cual significa que...

En la figura 8b se observa que conforme aumentamos el número de capas (layers), la precisión aumenta, excepto para 4 y 10 capas adicionales. Nótese que para epsilon > 0.8, la red con 6 capas adicionales tuvo el mejor desempeño. Aumentar el número de capas implica una sobreparametrización (overparameterization) de la red neuronal, lo cual significa que...

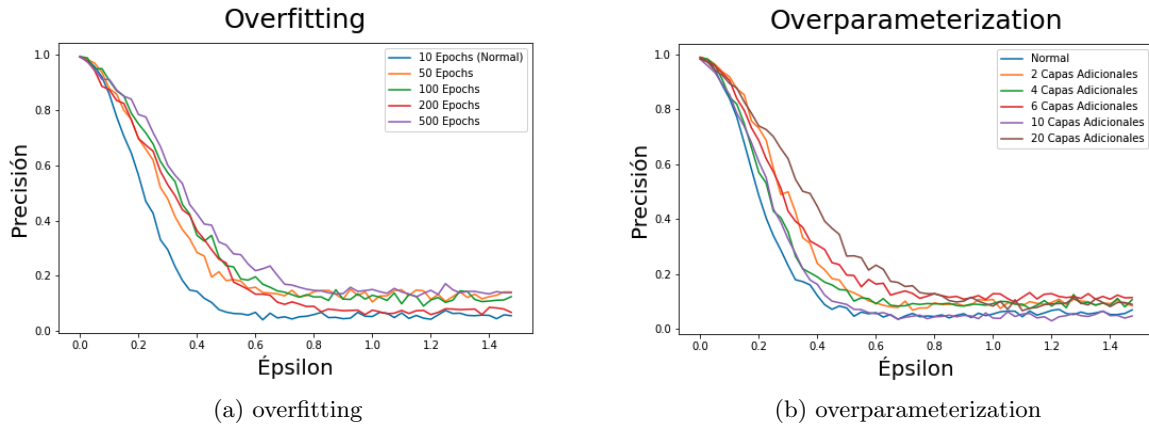


Figura 8: Efectos

- fix a large overparameterization and see if overfitting has the same effect
- pick an overparameterization and overfitting and check jpeg

### 3.4. Saliency

- show figures from notebook of how adversarial noise attacks parts of the image that seem vulnerable (for example changing a 3→8)
- Show Gradient-based localization of both image sets and how they change after adding adversarial noise[10]

## Referencias

- [1] Nicholas Carlini y David Wagner. *Towards Evaluating the Robustness of Neural Networks*. 2017. arXiv: 1608.04644 [cs.CR].
- [2] Nilaksh Das y col. *Keeping the Bad Guys Out: Protecting and Vaccinating Deep Learning with JPEG Compression*. 2017. arXiv: 1705.02900 [cs.CV].
- [3] Ian J. Goodfellow, Jonathon Shlens y Christian Szegedy. *Explaining and Harnessing Adversarial Examples*. 2015. arXiv: 1412.6572 [stat.ML].
- [4] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Inf. téc. 2009.
- [5] Alex Krizhevsky, Vinod Nair y Geoffrey Hinton. *The CIFAR-10 website*. 2009. URL: <http://www.cs.toronto.edu/~kriz/cifar.html>.
- [6] Yann LeCun, Corinna Cortes y CJ Burges. «MNIST handwritten digit database». En: *ATT Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist> 2 (2010).
- [7] Yann Lecun y col. «Gradient-based learning applied to document recognition». En: *Proceedings of the IEEE*. 1998, págs. 2278-2324.
- [8] Xingjun Ma y col. *Understanding Adversarial Attacks on Deep Learning Based Medical Image Analysis Systems*. 2020. arXiv: 1907.10456 [cs.CV].
- [9] Aleksander Madry y col. *Towards Deep Learning Models Resistant to Adversarial Attacks*. 2019. arXiv: 1706.06083 [stat.ML].
- [10] Ramprasaath R. Selvaraju y col. «Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization». En: *International Journal of Computer Vision* 128.2 (oct. de 2019), págs. 336-359. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01228-7. URL: <http://dx.doi.org/10.1007/s11263-019-01228-7>.
- [11] Uri Shaham, Yutaro Yamada y Sahand Negahban. «Understanding adversarial training: Increasing local stability of supervised models through robust optimization». En: *Neurocomputing* 307 (sep. de 2018), págs. 195-204. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2018.04.027. URL: <http://dx.doi.org/10.1016/j.neucom.2018.04.027>.
- [12] Uri Shaham y col. *Defending against Adversarial Images using Basis Functions Transformations*. 2018. arXiv: 1803.10840 [stat.ML].
- [13] Christian Szegedy y col. *Intriguing properties of neural networks*. 2014. arXiv: 1312.6199 [cs.CV].