Sistemas Cognitivos Artificiales

Roberto Casado Vara

Tema 11: Últimos avances en aprendizaje profundo

UNIVERSIDAD INTERNACIONAL DE LA RIOJA

La Universidad en Internet

En la clase hoy veremos…

- Generative adversal networks (GAN)
- Meta-learning y sus variantes
- Practica: GAN (fuente tensorflow)

Tema 11.1: Generative adversal networks

Un poco de historia

- Una de las áreas de investigación con más movimiento en la actualidad, dentro del mundo del aprendizaje profundo, es el de las generative adversarial networks (GAN), introducidas por primera vez por Ian Goodfellow en 2014.

- Las GAN son un tipo de modelo generativo. Los modelos generativos son una rama del aprendizaje no supervisado en la que, dado un conjunto de datos de entrenamiento, se intenta aprender la distribución de probabilidad de esos datos. A partir de esta distribución de probabilidad, podemos generar nuevos ejemplos similares a los de entrenamiento.

Un ejemplo

- Esto es similar a lo que vimos en los **modelos del lenguaje**. En ello considerábamos que el texto venía dado por una distribución de probabilidad en la que cada palabra depende de las palabras anteriores.

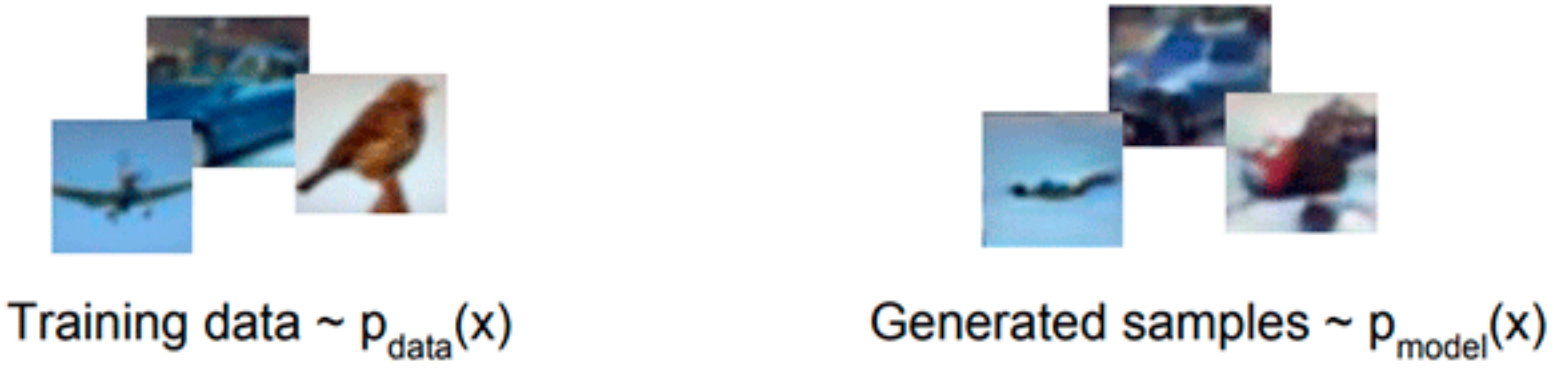


Figura 1: modelos generativos

Fuente: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture12.pdf

GAN

- En las GAN, a diferencia de estos últimos, no se intenta modelar una función de densidad (con la que posteriormente se pueden generar ejemplos), sino que directamente se aprende a generar ejemplos. Este tipo de modelo generativo se conoce como **implicit density estimation** (en oposición a explicit density estimation).

- La popularidad de las GAN se debe principalmente a sus impresionantes resultados, con modelos capaces de generar imágenes de gran realismo. Se aproximan al problema de la generación desde el punto de vista de un juego para dos jugadores. Estos dos jugadores son dos redes neuronales conocidas como **generator network** y **discriminator network**.

Juego de los 2 jugadores

- La **generator network** es una aproximación a la obtención de una muestra (sample) de la distribución de los datos de entrenamiento. En otras palabras: se encarga de generar imágenes parecidas al training set. Su input es un vector aleatorio, un vector que podemos cambiar a placer para generar imágenes distintas.

- La **discriminator network** es una red que intenta distinguir entre imágenes reales (pertenecientes a los datos de entrenamiento) y falsas (generadas por la generator network).

Generator y discriminator

- La **generator network** tiene como objetivo **engañar** a la **discriminator network** mediante la generación de imágenes que pasen por ser reales a ojos de esta.
- Por su parte, la **discriminator network** intenta **aprender** la diferencia entre imágenes reales y falsas para no ser engañada.
- Obtenemos una **red generadora** que ha aproximado el proceso de obtención de muestras a partir de la función de densidad de los datos de entrenamiento, pero sin modelar esta función directamente

Diagrama de una GAN

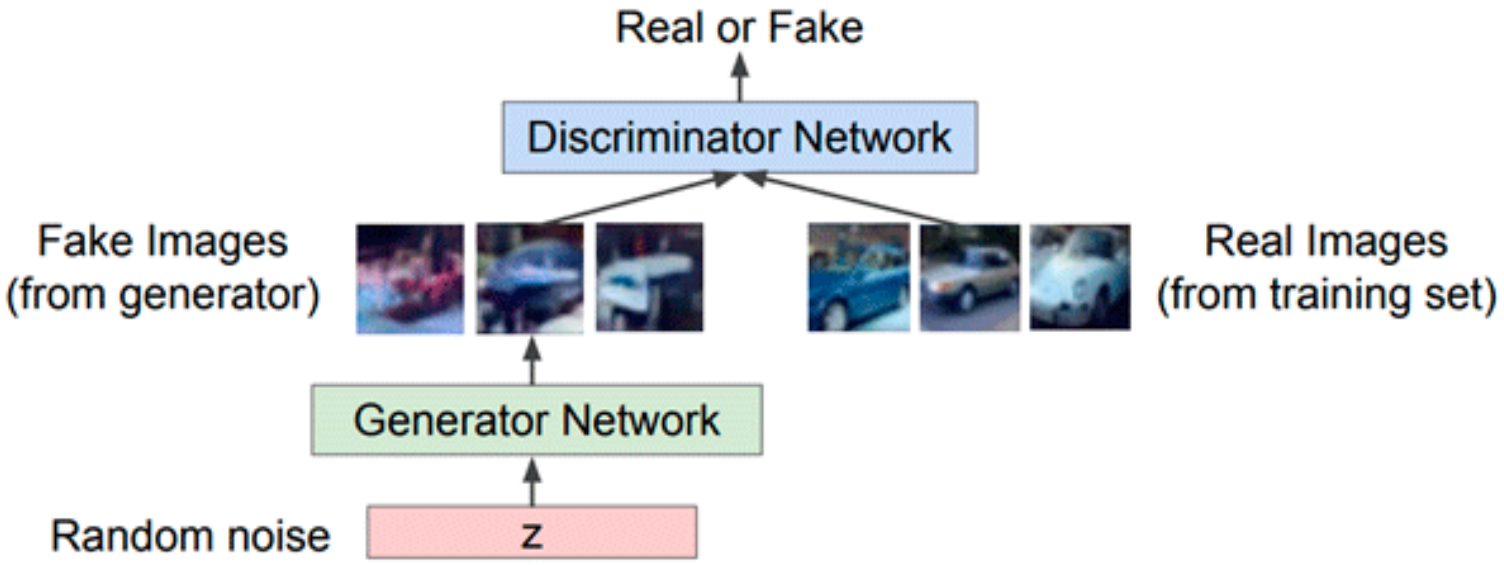


Figura 2: Generative adversal network

Fuente: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture12.pdf

Funcionamiento de una GAN

- Las dos redes son entrenadas a la vez. El entrenamiento o juego entre la generator y la discriminator network se plasma en la siguiente objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log \left(1 - D_{\theta_d} \left(G_{\theta_g}(z) \right) \right) \right]$$

- Donde:
  - D es la función del discriminador, con salida de un valor entre 0 y 1 (la probabilidad de que la imagen sea real).
  - G representa al generador (con salida de una imagen).

Funcionamiento de una GAN

- De este modo, el discriminador intenta maximizar el objetivo, de manera que:
- $D(\mathrm{X})$ sea lo más cercano posible a 1 (imagen real).
- Y $D(G(\mathrm{Z}))$ sea lo más cercano posible a 0 (fake).

- Por su parte, el generador intenta minimizar el objetivo, de manera que el discriminador piense que las imágenes generadas sean reales, esto es, que $D(G(\mathrm{Z}))$ sea cercano a 1.

Algoritmo original

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Figura 3. Proceso de entrenamiento seguido por Goodfellow.
Fuente: Goodfellow et al. (2014, p. 4).

Tema 11.2: Meta-learning

Idea inicial

- Meta-learning es otra área de investigación de gran interés para la comunidad de inteligencia artificial en la actualidad, si bien sus inicios datan de hace varias décadas.

- La idea básica de meta-learning es aprender y mejorar el proceso del aprendizaje o, en otras palabras, «aprender a aprender».
- **Objetivo** del meta-learning:
  - Los sistemas de inteligencia artificial «aprendieran a aprender» nuevas tareas de manera rápida usando su experiencia de aprendizaje anterior, en vez de considerar cada tarea de manera aislada.

Optimizer learning

- ¿por qué no aprendemos también el optimizador a utilizar, de manera que sea óptimo para que nuestra red aprenda de manera más rápida la tarea a resolver?
- En este tipo de meta-learning, hay una **red neuronal auxiliar**, conocida como **meta-learner**, que se encarga de aprender cómo hacer un update de los parámetros de la red principal (learner).
- La red meta-learner suele ser normalmente una **RNN**, lo cual facilita que esta recuerde cómo ha cambiado los parámetros en iteraciones anteriores.
- Esta red puede ser entrenada mediante aprendizaje supervisado o reinforcement learning.

Neural architecture search

- **Neural architecture search** trata el problema de diseñar una red como un problema de meta-learning: se intenta aprender a diseñar redes neuronales de manera que aprendan de manera eficiente distintas tareas.

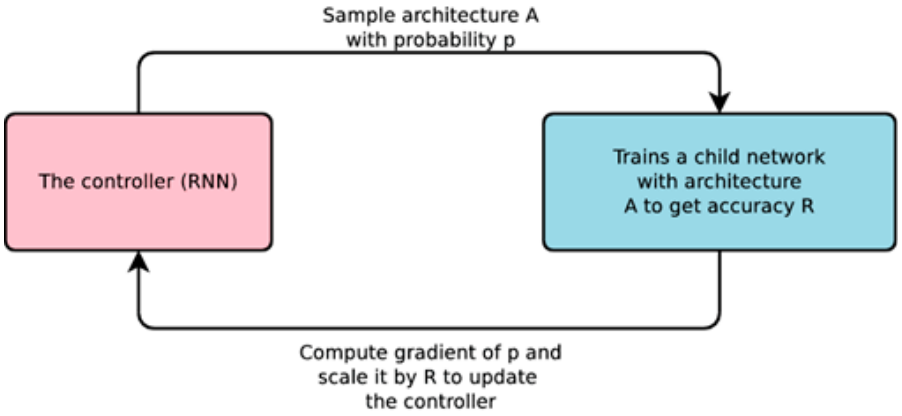


Figura 4: Ejemplo de neural architecture search

Fuente: https://ai.googleblog.com/2017/05/using-machine-learning-to-explore.html

Ejemplo: Neural architecture search

- Un ejemplo de este tipo de aproximamiento al meta-learning es AutoML de Google.
  - Una red neuronal recurrente (controller) se encarga de proponer un modelo (child network), que es entrenado y evaluado para una tarea.
  - El feedback recibido de esta evaluación se utiliza para informar al controller (siguiendo un marco de reinforcement learning) de cómo han afectado los cambios en la tarea.
  - Este proceso se repite miles de veces, permitiendo al controller diseñar modelos de mayor calidad para la tarea en cuestión.

Few shot learning

- En few-shot learning, el objetivo es conseguir que una máquina pueda generalizar conceptos y aprender a partir de un número reducido de ejemplos, tal y como haría un humano, y no a partir de datasets de gran tamaño.

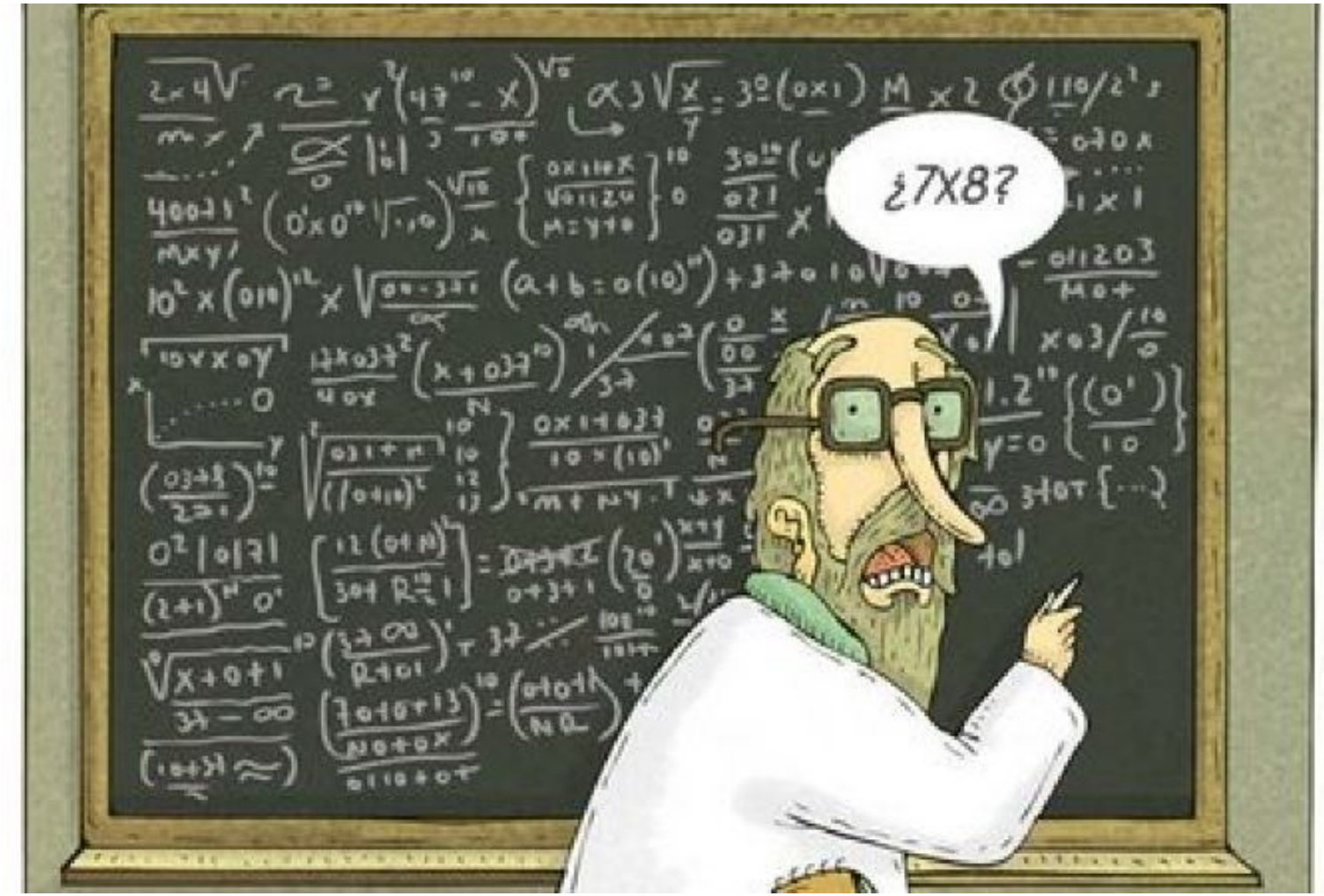
Figura 5: Ejemplo de Few shot learning

Fuente: https://bair.berkeley.edu/blog/2017/07/18/learning-to-learn/

Ejemplo practico: GAN

- Este ejemplo esta directamente extraído de las guías de tensorflow y no he querido modificarlo para que quede claro la fuente original y que no es código mío ni modificado por mi.

¿Dudas?



www.unir.net