Apunte clase 04/04/2024

1st Kelvin Núñez Barrantes
Ingeniería en Computación
Tecnológico de Costa Rica
San José, Costa Rica
kelvinnb@estudiantec.cr

I. Repaso

A. Perceptron

Propuesto por Frank Roserlant en 1957.Un Perceptron, de acuerdo con [1], se usan valores númericos de input con un peso y un bias para multiplicar los valores por los respectivos pesos y luego los productos son sumados junto al bias. Por último, se pasa por la función de activación y retorna un output."

Similar a una regresión logistica pero con otro loss function. Hinge Loss Se pensó que iba a resolver muchos problemas

Después se empezaron a mostrar problemas con los perceptrones por Minsky y otros. Tales errores fueron: 1. Requerimientos computaciones de la época 2.No puede resolver un XOR Debido a eso empezó el Invierno de la IA en donde se detuvieron las investigaciones

Al final, el problema del XOR se resolvió con las redes neuronales.

Estas nacieron como inspiración biológica de las neuronas que se conectan por las dendritas con otras neuronas para asì poder resolver problemas NO lineales.

B. Funciones de activación

La principal característica es que con la función de activación se decide si se deja pasar la información, la transforma o la bloquea. Existen varias funciones de activación.

C. Perceptrón Multicapa

La notación puede variar, las más comúnes son

$$W = \theta$$

$$\theta = (W, b)$$

$$XW$$

$$WX$$

El funcionamiento principal es que las neuronas manden la señal a todas las neuronas de la siguiente capa y así sucesivamente.

La primera capa se llama la Capa de entrada, tiene dimensión basada en la dimensión del X. Las siguientes capas se llaman Capas ocultas, de aqui se pueden poner N capas. La última capa se le conoce como la capa de salida.

La información se procesa de manera sucesiva entre las capas, la información que suelta una capa la tira a la siguiente hasta llegar a la capa final. A este proceso se le conoce como deepforward. Esto sirve para hacer la predición del modelo.

El proceso de entrenamiento del modelo se le conoce como backpropagation. Se hace la propagación del error hacia atrás para poder actualizar todos los parámetros.

Lo importante es decidir cuál es la mejor función de activación que se requiere dependiendo de la tarea que se está haciendo.

Con respecto a la capa de salida, esta depende de los resultados de la capa anterior.Al terminar todas las capas se puede sacar el Loss function y así podemos optimizar los parámetros W y b. Se utiliza de la siguiente manera:

$$\frac{\partial L}{\partial W_i^j}$$

El problema es que hay que calcular esa derivada parcial para optimizar los valores.

A mayor cantidad de capas y mayor cantidad de neuronas, se va a tener un mayor rendimiento pero va a ser más difícil entrenar el modelo. Al meter capas de procesamiento de datos se empieza a hablar de Deep Learning

D. Maldición de la dimensioonalidad

Entre más features se agregue a la red neuronal, va a ser dificil hallar ese punto óptimo. Aumenta la computabilidad por lo que también puede ser difícil encontrar patrones en los datos. Para resolver esto se pueden utilizar técnicas para reducir la dimensionalidad

Existe un comportamiento jerárquico en donde:

- Al hacer el analisis se aprenden cosas simples para realizar algo más complejo.
- Ganancia exponencial en comportamiento de polinomios.
- Se pueden hacer aproximaciones de funciones con Reds Neuronal.
- Composición basada en el aprendizaje.

II. Funciones de activación

Son las funciones que se colocan inmediatamente después de la regresión lineal. A la regresión lineal también se les llama preactivación

Se utilizan porque estas logran agregar un comportamiento específico a la red.

Permite controlar el rango de salida: Deja pasar o no la señal.

A. *Función lineal*

No funciona porque al calcular el descenso del gradiente y optimizar parámetros la derivada no depende de la entrada, el gradiente es el mismo después de cada iteración al ser lineal.

B. *Sigmoide*

Función de activación que da resultados entre 0 y 1. Siempre es positiva y estrictamente creciente.

El problema es que la derivada termina dando resultados muy cercanos al cero por lo que al optimizar los valores de a y b el ritmo de actualización va a ir disminuyendo. Llamado **Vanishing gradient**

C. *Tangente hiperbólica*

La activación está entre -1 y 1. Positivos y negativos centrados en el origen. Estrictamente creciente y se utilizó mucho en modelos de lenguaje.

Es más fácil converger cuando los valores están más centrados. Mismo problema de la Sigmoide.

D. *Rectifier Linear Unit(ReLU)*

Acotada debajo del cero No está acotada en los positivos.

Estrictamente creciente y más eficiente para el Deep Learning y manejo de las capas ocultas.

Mata las activaciones. Los logits son la entrada que están antes de la función de activación.

El problema es que NO es derivable para el valor del 0. Aunque es improbable que un resultado dé un valor tan específico, no presenta un problema tan grave por la poca posibilidad de que suceda. Además, es posible que genere neuronas muertas que son neuronas que no sirven para el siguiente comportamiento

E. *Leaky ReLU*

Resuelve el problema de las neuronas muertas haciendo una multiplicacion por una constante.

Es probable que no sea la mejor opción para todos los casos ya que la pendiente puede ser

F. *Parametric ReLu*

Otra solución para las neuronas muertas.Aprende un parámetro que va manejando la pendiente de la parte negativa El parámetro es otro valor a aprender en la red.

G. *References*

[1] Bhardwaj, A. What is a Perceptron? - Basics of Neural Networks. [Online]. https://towardsdatascience.com/what-is-a-perceptron-basics-of-neural-networks-c4cfea20c590