Apuntes Semana 8 - 08/04/2025

Jose Carlos Umaña Rivera
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica

***Abstract*—Este documento recopila los apuntes correspondientes a la clase del dia. Se presentan respuestas al Quiz 4. Además, se incluyen observaciones generales sobre la primera tarea y un repaso teórico sobre perceptrones, redes neuronales, y funciones de activación. Se continúa con el tema de funciones de activación con funciones como ReLU, Sigmoid, Tanh, Leaky ReLU, PReLU y Softmax.**

I. Respuestas Quiz 4

A. Describa la técnica de Normalización y Estandarización

1) Normalización:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

2) Estandarización:

$$x' = \frac{x - \mu}{\sigma}$$

B. Anote la derivada de la función sigmoide:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

$$\sigma'(x) = \sigma(x)(1 - \sigma(x))$$

C. Describa las fórmulas para calcular el Accuracy, Recall, Precision y F1-Score

1) Accuracy:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

2) Recall:

$$\text{Recall} = \frac{TP}{TP + FN}$$

3) Precision:

$$\text{Precision} = \frac{TP}{TP + FP}$$

4) F1-Score:

$$F1 - Score = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

D. Dada la función de verosimilitud, desarrolle la log-verosimilitud en 3 pasos

$$\mathcal{L} = \prod f_{w,b}(x_i)^{y_i} \left(1 - f_{w,b}(x_i)\right)^{1-y_i}, i = 1...N$$

$$\ln(\mathcal{L}) = \sum \ln\left(f_{w,b}(x_i)^{y_i} \left(1 - f_{w,b}(x_i)\right)^{1-y_i}\right)$$

$$\ln(\mathcal{L}) = \sum \left[y_i \ln(f_{w,b}(x_i)) + (1 - y_i) \ln(1 - f_{w,b}(x_i))\right]$$

II. Feedback de la tarea 1

En general se comentó que los graficos estan bonitos, pero se falla en sacar conclusiones de estos, por ejemplo en el caso de los boxplots. En el proyecto se espera que todas las imágenes estén referenciadas y no que simplemente hayan gráficos tirados por el documento. También se habló que es necesario mostrar las pruebas de diferentes hiperparámetros para observar el progreso obtenido para llegar a los elegidos y no solo mostrar el resultado de los obtenidos. Por último se pidió observar y poner atención a los graficos de validación y que estos tengan un comportamiento esperado.

III. Anuncios

Hay 2 lecturas nuevas para el proximo quiz: Capitulo 6 (se habla de un tema importante como el vanishing gradient) y el paper sobre arquitectura para computer vision (ImageNet Classification with Deep Convolutional Neural Networks).

IV. Repaso

A. Perceptron

El perceptron es como una regresión logísstica pero diferente (otro loss function). Llega el invierno de la IA, se presenta el error del XOR, pues este no puede ser modelado de una regresión logística.

B. Redes neuronales

Resuelven el problema del XOR. Tienen una inspiración biológica de las neuronas, pues similar a las dentritas en el núcleo se hacen las ”matemáticas”.

C. Función de activación

En regresión logística se le llamaba función no-lineal (sigmoide), esta dependiendo de la señal activa o no activa la neurona, tambien es la encargada de dejar pasar, transformar o bloquear la información. Existen muchas de estas como:

1) Perceptrón Multicapa (MLP): Esta es capaz de resolver un XOR con 2 neuronas. Realizandose para todas las capas y donde la capa siguiente necesita de la anterior.

2) ¿Está bien usar sigmoid siempre?: Respuesta general: NO. Si no ejecutar según la que se utilice en la primera capa.

3) ¿Cuantas capas o cuantas neuronas por capa?: A base de experimentación se sabe cuantas capas o cuantas neuronas.

4) Maldición de dimensionalidad: Entre mas dimensionalidad, más complejo el problema, se recomienda intentar minimizar la cantidad de features.

V. Funciones de activación

Como se comentó anteriormente, agrega comportamiento a la red, permitiendo controlar el rango de salida y deja pasar la señal a la siguiente capa. Hay muchas funciones que se pueden utilizar, estas buscan transformar los valores de entrada.
Sin una función de activación se tendra un hiperplano infinito y con una se cambiará ese comportamiento.

A. Función Lineal

Se podría dejar así, pero esto trae problemas pues la derivada no depende de la entrada, entonces el gradiente será el mismo para cada iteración, no aprende nada de los datos.

B. Sigmoide

Tiene una activación entre 0 y 1, siempre es positiva, es acotada y es estrictamente creciente. Entre sus problemas tiene que la derivada es cercana a 0, teniendo un gradiente pequeño (al final de la función), entonces el entrenamiento se estanca o converge lentamente, problema también llamado "vanishing gradient".

C. Tangente Hiperbólica

Tiene una activación de entre -1 y 1, teniendo positivo y negativo, entonces es como la sigmoide, pero esta esta centrada en el origen, siendo estricamente creciente, diferenciable en cualquier punto y limitada. Suele ser utulizada en modelos de lenguaje (LSTM). Presenta los mismos problemas que la sigmoide.

D. Rectifier Linear Unit (ReLU)

$$g(x) = \max(0, x)$$

Esta acotada debajo del cero, es estrictamente creciente, eficiente para Deep Learning y mata las activaciones. Tiene un problema marcada, como lo es el de las neuronas muertas, pues no es derivable en todos los puntos, ademñas de que el gradiente llega a cero en algunos casos, entonces los pesos no son actualizados.

E. Leaky ReLU

Asigna una pequeña constante al mínimo permitido, resolviendo el problema de las neuronas muertas. Es un buen intento, pero no el mejor.

$$g(x) = \begin{cases} 0.01, & x < 0 \\ x, & x \geq 0 \end{cases}$$

$$\frac{\partial g(x)}{\partial x} = \begin{cases} 0.01, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

F. Parametric ReLU

Permite aprender un parámetro para dejar que la señal continúe, este es aprendido al igual que el resto de la red.

$$g(x) = \begin{cases} w, & x < 0 \\ x, & x \geq 0 \end{cases}$$

$$\frac{\partial g(x)}{\partial x} = \begin{cases} w, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

G. Softmax

La función convierte el outputlayer en una distribución de probabilidad. La sumatoria normaliza el output layer.

$$\sigma(x)_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}}$$

Normalmente usada como salida de clasificación. El verctor de entrada es llamado logits. Tiene el Cross Entropy Loss.

1) ¿Por qué usar e^x?: Porque es estrictamente creciente y evita valores negativos.

2) Cross-Entropy Loss: Se usa como Loss Function en softmax. También conocida como Log-Loss o Logistic Loss, representa probabilidades en un espacio logaritmico [0, 1] y es numericamente estable.

$$\mathcal{L}_\rangle = \log(P(Y = y_i, X = x_i))$$

$$\mathcal{L}_\rangle = -\log(\frac{e^{s_k}}{\sum e^{s_j}}), conjdesde 1, .., C$$

H. ¿Cual Función de Activación utilizar?

Depende mucho del problema, las funciones Sigmoid y Tanh tienen el problema del "vanishing gradient", por lo que se recomienda comenzar con ReLU, si no con otra, esto debido a su rapidez de computar y su alto uso en DeepLearning, si no continuar con Leaky ReLU o Parametric ReLU.