NOMBRE: Bernardita Rosas

SECCIÓN:

PUNTAJE:

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2613 — Inteligencia Artificial — 2' 2023

Tarea 5 – Respuesta Pregunta 1

1.1. Comprendiendo los datos Investiga y describe por qué el preprocesamiento es una etapa crucial para las redes neuronales. Explica cómo los pasos de preprocesamiento pueden afectar el rendimiento de un modelo.

De acuerdo a un estudio (Peralta et al., 2016), por un lado, el preprocesamiento resulta fundamental al buscar corregir errores en los datos que puedan dañar el aprendizaje, omitir ruidos y valores extremos, asegurar consistencia, y en otros casos, adaptar los datos que se buscan trabajar al problema particular. Por otro lado, puede reducir costos tanto de tiempo como de recursos al simplificar problemas, sobre todo a gran escala, facilitando el encuentro de patrones. Además, muchas veces, los algoritmos de machine learning no pueden ser ocupados directamente en los datos por lo que el preprocesamiento cumple una tarea muy importante en tal caso.
En cuanto a cómo los pasos de preprocesamiento pueden afectar a los modelos, es importante considerar que estos procesos deben adaptarse según el modelo que se desea entrenar. Por ejemplo, según (Peralta et al., 2016), la propuesta de centrado de imágenes facilita el entrenamiento de la red FCDNN, mientras que combinaciones más complejas funcionan mejor con LeNet. Entonces, es importante utilizar distintas técnicas de preprocesamiento para así lograr descubrir la que se adecúe de mejor manera al modelo, problemas y datos en su conjunto.

Si tuviéramos pocas imágenes para entrenar un modelo, ¿qué técnicas podrías usar para enriquecer el conjunto de datos? Investiga sobre el aumento de datos en imágenes.

De acuerdo nuevamente a un estudio sobre el preprocesamiento para redes neuronales profundas (Peralta et al., 2016), el aumento de datos vía transformación de los datos originales es utilizado para enriquecer el conjunto de datos al introducir transformaciones (traslaciones, rotaciones, simetrías, deformaciones elásticas, etc...), de esta manera se puede reducir el ruido y sobreentrenamiento, al mostrarle al modelo que tales cambios no afectan el resultado y aumentar la variabilidad de los datos.

1.2 Explorando los Multilayer Perceptron (MLP)
Discute cómo la profundidad y el ancho (número de capas y neuronas por capa) de un MLP afectan su capacidad para aprender patrones complejos. ¿Cuáles son los desafíos relacionados con el aumento de la complejidad del modelo?
De acuerdo con esta tesis (Jiménez, 2012), las capas ocultas representan relaciones entre las variables de entrada que no pueden interpretarse a simple vista, con ello al aumentar la profundidad de esta, aumenta su capacidad de aprender patrones complejos conforme aumenta su cantidad de parámetros al aprovecharse de la dependencia entre las variables. Adicionalmente, se establece que el número de capas ocultas corresponde a un parámetro a optimizar al tener en cuenta el costo computacional que implica.
En cuanto a la profundidad, es fácil ver que la cantidad de capas afecta la capacidad de aprender patrones

complejos al permitir una abstracción mayor y con ello tomar en cuenta detalles más sútiles, aunque dificultando el entrenamiento. Por lo mismo, cuando se aumenta la cantidad de parámetros también puede aumentar el sobreajuste al tomar en cuenta factores que pueden no ser tan importantes. Generalmente con una capa oculta es suficiente tesis (Gónzález, S/F) y en muchos casos agregar una mayor cantidad de capas ocultas tan solo ralentaliza el resultado.
En cuanto al ancho, si este es mayor, entonces se pueden analizar una mayor cantidad de factores para tomar decisiones. Cabe destacar que, en el caso de la capa de entrada, este depende de los datos iniciales que se dispongan mientras que la cantidad de neuronas en la capa de salida viene dado por la cantidad el tipo de análisis que realizamos (y sus variables). Generalmente, un aumento de neuronas ocultas aumenta la eficacia del aprendizaje. Según esta tesis (Gónzález, S/F), en tal ejemplo se paso de 4 a 6 neuronas por capa y esto hizo que no solo el modelo mejorara su rendimiento sino que también aceleró el proceso.
Los desafíos principales al hablar de aumentar la profundidad y/o ancho, y con ello la complejidad del modelo, tienen que ver con el sobreentrenamiento y el aumento del tiempo de ejecución, además de la disminución de la interpretabilidad. Cabe notar que no siempre mejora el rendimiento del modelo por lo que es necesario probar distintas combinaciones que resulten de la mejor manera.

Investiga y compara al menos dos funciones de activación no lineales diferentes utilizadas en MLPs. ¿Cómo afectan estas funciones al tipo de decisiones que puede aprender la red? Considera aspectos como la saturación y la no linealidad.

En primer lugar, según la tesis (Jiménez, 2012) consideraremos la función de activación sigmoidal. Esta tiene la siguiente fórmula: $\varphi(v) = \frac{1}{1+e^{-av}}$, es siempre creciente, derivable y toma su valor máximo cuando v es cero, por lo que se pueden aplicar técnicas aplicables a la función escalón, con la ventaja de que su derivada si está definida (además de que siempre es cercana a 0 para valores grandes negativos o positivos). Es propensa a la saturación para entradas muy grandes o muy pequeñas (por lo que es más propensa al sobreajuste) y no es lineal por lo que puede representar distribuciones no lineales.
En segundo lugar, según la tesis (Jiménez, 2012), también se encuentra la función de activación Gaussiana de función: $\varphi(v) = A \cdot e^{-Bv}$, que tanto la posición de los centros como la anchura pueden ser configuradas, de manera que son más adaptativas que las funciones sigmoidales, por lo que, en algunos casos, presentan ventajas en modelos más complejos. Una ventaja de las sigmoidales es que al tener siempre un valor entre 0 y 1, resulta más simple modelar probabilidades si es que el modelo lo requiere o se tiene un problema de clasificación, además de que se reduce el espacio de posibilidades al acotar el rango. Es menos propensa a la saturación que la función sigmoide, aún puede sufrir de desvanecimiento del gradiente.
De todas formas, es importante notar la importancia de considerar el problema particular en la elección de la función de activación del MLP.

Examina las causas y síntomas del overfitting en MLP. ¿Qué técnicas se pueden aplicar para prevenir este problema y cómo afectan el proceso de aprendizaje?

De acuerdo a (Jimenez, 2012), "cuando el número de parámetros es igual al número de muestras, la solución de mínimos cuadrados puede conseguir un ajuste perfecto a los datos de aprendizaje mientras ofrece una generalización muy pobre respecto a muestras nuevas", es decir, ocurre sobreajuste al aumentar parámetros que se evidencia en que el accuracy mejora en cuanto a los datos de entrenamiento, pero no en los datos de testeo ya que el modelo es demasiado complejo por lo que se enfoca en ruido, y no así en los factores determinantes. Otra razón que puede causar sobreajuste es una cantidad limitada de datos.
Este puede disminuirse ya sea mejorando el preprocesamiento (según el cual los datos incluyen una menor cantidad de información irrelevante), cuyo impacto en el aprendizaje viene dado en que lo simplifica al tener que analizar menos y mejores datos, o disminuyendo la complejidad (parámetros del modelo), pero es nece-

sario tener cuidado con capturar el comportamiento inherente del conjunto de datos, ya que, de lo contrario, se llega a subajuste, su impacto en el aprendizaje nuevamente implica que se simplifica el modelo al considerar un nivel menos de abstracción y menor cantidad de parámetros. También se pueden usar modelos bayesianos que permite modelos más flexibles y por ello trabajar eficientemente en modelos más complejos, o por supuesto, aumentar (y diversificar) la cantidad de datos que le da más información al modelo para mejorar su aprendizaje.

Más allá del SGD (Stochastic Gradient Descent) y Adam, investiga sobre otro método de optimización utilizado en el entrenamiento de MLP. ¿Cuáles son sus características únicas y en qué situaciones podría ser preferible?

A continuación explicaré RMSprop según el artículo (Bushaev, 2018). Su característica más importante corresponde a que se adapta el learning-rate individualmente para cada peso, dividiendo la tasa de aprendizaje para un parámetro particular por la raíz cuadrada del promedio cuadrático de los gradientes anteriores para ese parámetro (asegurándose así de que no varíe tanto las tasas de aprendizaje en puntos cercanos), por lo que se reduce la posibilidad de llegar a puntos sillas que no resultan interesantes en la optimización. Con RMsprop, permitimos que una mayor estabilidad en el entrenamiento pero no se permite que el estimado se vuelva en sobreajuste al mantener un promedio móvil. De esta forma, si los gradientes son relativamente grandes, entonces el tamaño del paso será "pequeño" y con esto se busca evitar oscilaciones alrededor del mínimo global, y viceversa (Sotáquira, 2021). Por todo lo anterior, es que este algoritmo es preferible en problemas no convexos o espacios complicados con cambios bruscos de gradiente al lograr converger más rápidamente.

Identifica y analiza un estudio de caso donde se haya utilizado un MLP para resolver un problema real. ¿Qué características del problema hicieron que un MLP fuera una buena elección y cómo se diseñó la arquitectura de la red para adaptarse a las necesidades específicas del problema?

Según el caso (Díaz et al., 2009), la estimación de la Tensión de Ruptura por termofluencia (creep) en aceros ferríticos, los MLP resultaron idóneos dado que corresponden a un problema muy importante de la ciencia de los materiales que no ha podido ser resuelto adecuadamente desde los modelos físicos-matemáticos al no existir relaciones evidentes. En particular, se menciona que MLP demostrado ser superior a los modelos anteriores en la extrapolación y representación de datos de creep.
Se probaron distintas combinaciones para observar bajo qué arquitectura se obtenían los mejores resultados y se llegó a una cantidad de 11 neuronas en la capa oculta (una), 37 neuronas en la capa de entrada (cantidad de características a observar) y tan solo una neurona de salida conteniendo la estimación del creep. Además, en el entrenamiento del modelo se utilizaron los algoritmos BackPropagation y PSO. Esta arquitectura es adecuada para un problema que tan solo devuelve un resultado pero toma en consideración 37 factores, mientras que la forma más certera de saber si las combinaciones son correctas implica experimentos empíricos. En cuanto al uso de PSO, se expresa que "implementar la reinicialización frecuente del vector velocidad hacia valores aleatorios en el algoritmo clásico de PSO induce una gran eficacia en el algoritmo pues evita consistentemente, la convergencia prematura hacia mínimos locales" (Díaz et al., 2009), denostando su utilidad en el problema.

1.3 Implementación de un Multilayer Perceptron (MLP)

Evalúa el rendimiento de tu modelo en el conjunto de datos de prueba. Utiliza métricas relevantes para problemas de regresión, como el error cuadrático medio (MSE).
Training set score: 0.373412
Test set score: 0.347225

Mean Squared Error (MSE): 86816002020.927460
R^2 Score: 0.347225
Mean Absolute Error (MAE): 216567.291456
Mean Absolute Percentage Error (MAPE): 37.798526%

Realiza al menos dos experimentos variando la arquitectura del MLP o los hiperparámetros. Describe cómo cada cambio afecta el rendimiento del modelo y discute tus hallazgos. Comenzamos aumentando el learning rate de 0.1 a 0.2 y notamos las siguientes medidas: Training set score: 0.373184
Test set score: 0.357966
Mean Squared Error (MSE): 85387519463.787888
R^2 Score: 0.357966
Mean Absolute Error (MAE): 216030.249979
Mean Absolute Percentage Error (MAPE): 37.410590%
Lo más notable corresponde al aumento del Test set score o R^2, además de que se disminuyó notablemente la cantidad de iteraciones en las que se logró convergencia indicando que, a pesar de ser un modelo más simple, este tiene una capacidad de generalización mayor. Aún así, hay que tener cuidado con aumentar demasiado el learning rate, ya que, nuevamente, puede que el modelo disminuya su rendimiento.

A continuación, aumentaremos la complejidad del modelo al cambiar las neuronas por capa 4 a 8 y la cantidad de capas de 2 a 3.
Training set score: 0.353122
Test set score: 0.320752
Mean Squared Error (MSE): 90336692045.026474
R^2 Score: 0.320752
Mean Absolute Error (MAE): 213915.308712
Mean Absolute Percentage Error (MAPE): 34.799276%
Podemos notar que el modelo rindió de manera notablemente peor luego de que se complejizará, demostrando que no siempre es beneficioso el aumento de parámetros y la complejidad. Definitivamente, es necesario encontrar un punto óptimo en cuanto a complejidad.

Basado en tus resultados, ¿cuáles son las características más importantes de un MLP para la predicción precisa en este conjunto de datos? ¿Cómo relacionas esto con la teoría aprendida? En base a los experimentos que expusé y también otros que hice por mi cuenta (aumentando cantidad de neuronas por capa y cantidad de capas ocultas), cambios en solvers, escalamiento y funciones de activación, se puede establecer que existen muchos factores importantes y es necesario probar distintas combinaciones para encontrar la óptima. En el caso de la arquitectura, encontré que ambas capas con 4 neuronas era óptimo ya que cualquier combinación de más neuronas presentaba señales de sobreajuste (también más capas) mientras que menos neuronas empeoraba notablemente el rendimiento al presentar un subajuste. En cuanto a funciones de activación, nuevamente, la decisión era excesivamente importante al hacer la diferencia entre un score de 37% y 1%, lo que yo realmente no esperaba que fuera a afectar tanto.

Otra cosa importante que note es que por ejemplo al pasar de stochastic gradient descent a adam, las mismas funciones de activación, arquitectura u otros no eran necesariamente los mismos óptimos para el modelo, incluso podían ser completamente opuesto que resalta la importancia de probar distintas combinaciones para lograr los resultados esperados.

En cuanto a la relación con la teoría aprendida, definitivamente se cumple lo que comentó acerca del sobreajuste del modelo y sus posibles causas relacionadas con la cantidad de capas y de neuronas por capa, pero al mismo tiempo es necesario optimizar tales valores sin caer en el subajuste. Otro factor importante aprendido tiene que ver con la necesidad de probar diversas funciones de activación para probar la mejor para el problema dado.

Describe los desafíos que encontraste al implementar y entrenar el MLP. ¿Cómo los superaste y qué aprendiste en el proceso? ¿en qué otros tipos de problemas crees que un MLP podría ser efectivo? Tuve hartos problemas por no darme cuenta de que el modelo que se hizo en clases era de clasificación mientras que este era de regresión lineal por lo que perdí tiempo en esto. Lo superé al darme cuenta y adecuar lo que llevaba. No sé si aprendí cómo tal pero reforcé la idea que siempre he tenido de que si algo no funciona debes verlo desde una perspectiva y empezar de nuevo de ser posible.

Otro problema que tuve fue ya que no sabía que hacer con las columnas no numéricas pero luego por el grupo mencionaron que se utilizaba onehotencoder con lo que me ayudo mucho. En definitiva, la comunicación a través de los grupos de telegram fue muy útil.

Me parece que un MLP podría ser útil para predecir desempeño académico en base a notas del colegio, puntaje de entrada o algo así, para mí sería interesante ver cuales son los factores más decisivos para un buen desempeño académico en la universidad y quizás cambiar ciertas ponderaciones.

1.4 Introducción y teoría de las CNN's ¿Qué es una operación convolucional? ¿Qué es un kernel? Utiliza estos conceptos para explicar el rol de las capas convolucionales en una CNN.

De acuerdo con el artículo (LearnOpenCV, S/F) provisto, se tiene que primero es importante considerar el que se le aplica un filtro en el que se cambia la dimensionalidad de los input de la red. Luego el conjunto de tales filtros que se aplican a los inputs es referido como kernel (matriz de pesos). Por otro lado, una operación convolucional consiste en colocar el kernel sobre una porción del input, aplicando multiplicaciones y sumas ponderadas en regiones locales. A partir de esto, el rol de las capas convolucionales en una CNN corresponde a utilizar múltiples kernels para así aprender características complejas ya que cada filtro contiene pesos que se han aprendido durante el proceso de entrenamiento, almacenando sus resultados obtenidos a partir de la capa anterior en mapas de activación.

¿Cuál es el rol de las funciones de activación? ¿Y de las capas de Max Pooling?

Las funciones de activación se utilizan luego de que ya se aplicaron los filtros en las capas de convolución y se produjo un número. Tal número luego se pasa a la función de activación, según la cual se produce el output que pobla el mapa de activación para cada una de las neuronas.
Luego, el rol de las maxing pools corresponde a disminuir la dimensión espacial de los datos, así luego de una serie capas convolucionales, se utilizan para disminuir el espacio espacial de los mapas de activación, reduciendo la cantidad de parámetros y con ello el sobreajuste y el poder computacional necesario (LearnOpenCV, S/F).

Quizás habrás notado que la mayoría de arquitecturas de CNN's utilizan una última capa conocida como flatten layer. ¿Cuál es su función? ¿Cuál es el rol de la función softmax en ella?
De acuerdo con (education.io, S/F), su función principal es transformar los mapas de características tridimensionales generados por las capas convolucionales y max pooling en una forma que pueda ser comprendida por las capas densas (una dimensión).

En cuanto a la función softmax, esta se encarga de tomar los valores que arroja la flatten layer y transformarlos en probabilidades para la output layer para expresar la confianza de la red en cada clase de salida

posible.

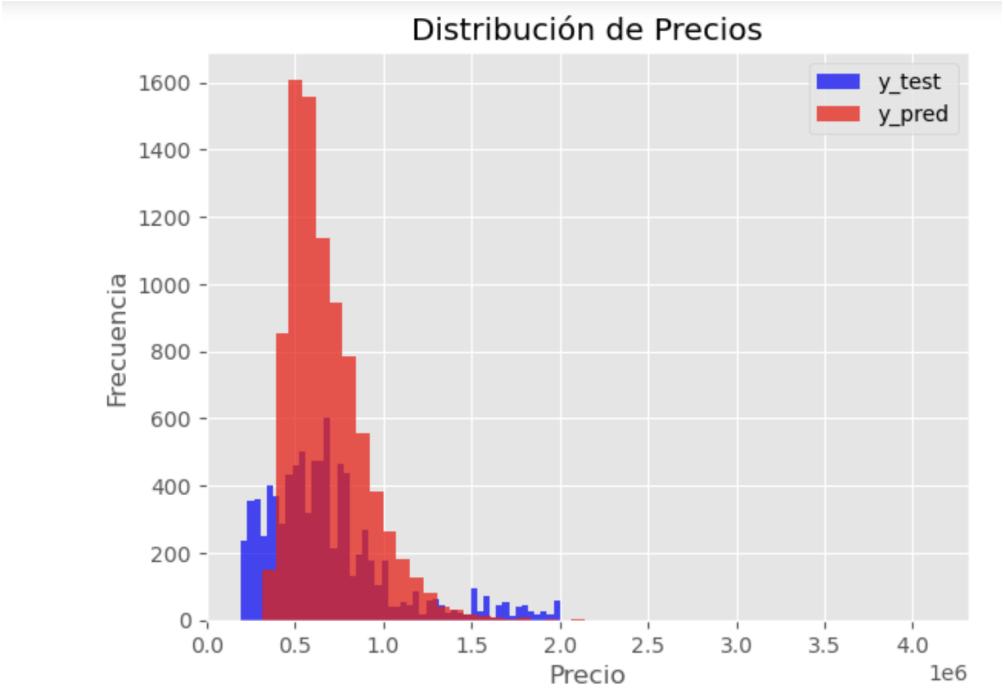
¿Cuáles son las ventajas y desventajas de las CNN's frente a las MLP's? ¿Para qué tipo de tareas suele ser útil utilizar CNN's?

La ventaja principal corresponde a que los CNN's utilizan una menor cantidad de parámetros por lo que se evita el sobreajuste ya que no es utiliza una neurona por cada píxel de una imagen, sino se aplican filtros. Por lo mismo, también las CNN's requieren de una menor cantidad de tiempo de entrenamiento, otro elemento muy importante. Adicionalmente, los MLP tienen problemas en identificar imágenes que han sido deformadas, produciendo resultados no tan confiables (LearnOpenCV, S/F). En cuanto a desventajas de CNN's, estas no son muchas pero se puede considerar el que es mejor que se use cuando los inputs son más complicados ya que los MLP's funcionan mejor en casos más simples. Sobre todo, en cuanto a procesamiento de imágenes que se tienen inputs de mayor dimensionalidad, CNN's son especialmente útiles por su reducción de parámetros.

1.5 Creando y evaluando una CNN El accuracy corresponde alrededor de un 55%, lo que es bastante bueno considerando que se realizaron a partir de tan solo una foto.
En cuanto a la función de pérdida, podemos observar que el sobreajuste no es demasiado ya que ambas funciones de pérdida son parecidas en cuanto a rendimiento, aunque la de validación oscila más que la de entrenamiento. Sirve justamente para ver intuitivamente la gravedad del sobreajuste.

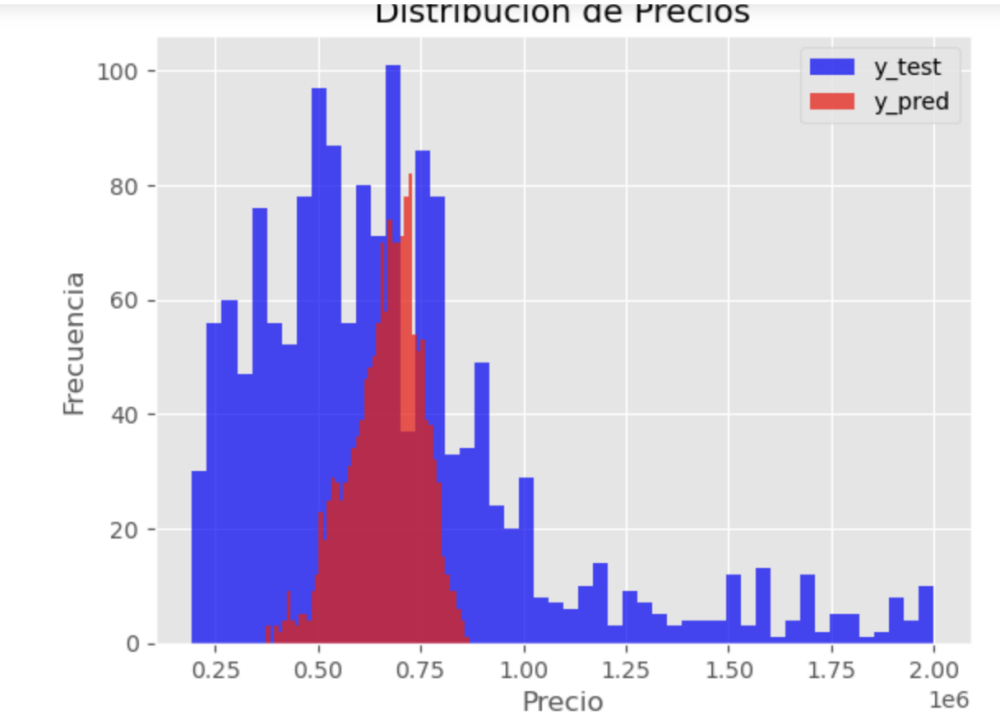
1.6 ¿Cuál de los modelos tiene un mejor accuracy?
Es evidente que el modelo de MLP presenta mejor rendimiento, lo que posiblemente se debe a su simplicidad ya que el de CNN es mucho más complejo. Comparando MSE, MLP presenta 86.816.002.020(accuracy=62%) mientras que CNN presenta 90.336.692.045(accuracy=56%), una diferencia considerable aunque no tan grande como quizás hubiese esperado dadas sus diferencias en accuracy. Como ya expresé, me parece que la principal razón de esto tiene que ver con la complejidad del segundo modelo.

¿Qué rangos de precios tienen un mejor rendimiento? ¿Y cuáles el peor? ¿Depende del modelo?
Insertaré imagenes de la distribución de ambos modelos:




```
Media y_test: 685787.38
Mediana y_test: 620000.00
Desviación Estándar y_test: 364685.13
Media y_pred: 665860.88
Mediana y_pred: 678629.19
Desviación Estándar y_pred: 87807.57
```

MLP




```
Media y_test: 679828.25
Mediana y_test: 613000.00
Desviación Estándar y_test: 354172.96
Media y_pred: 665860.88
Mediana y_pred: 678629.19
Desviación Estándar y_pred: 87807.57
```

CNN

Claramente, los rangos de precios más cercanos a la mediana son los que presentan mayor accuracy ya que también el modelo tiene un sesgo al tener más datos respecto a estos en relación a datos más aislados. Los datos aislados presentan peor desempeño. Sí depende hasta cierto punto del modelo, podemos notar un menor sesgo en el MLP mientras que CNN tiene una menor dispersión, pero me parece que el problema principal se encuentra en los datos.

2. DCCanario

Actividad 1: Comprendiendo los hiperparámetros

¿Cuál es la función del exploration rate?

De acuerdo con Makone (2021), el exploration rate tiene que ver con que tantas nnuevas posibilidades de obtener mejores resultados. Una alta taza de exploración implica probar disintas acciones que quizás lleven a una mejor o a una peor recompensa, a un costo computacional. Lo opuesto corresponde a exploitation, según la cual el modelo se asegura de una buena recompensa pero no explora si es que existen mejores. El balance entre ambos es muy importante para asegurar el buen rendimiento del modelo.

¿Cuál es el problema de tener un exploration rate mínimo con un valor muy bajo y un exploration decay rate con un valor muy alto?

Luego, exploration decay rate es que tan rápido bajamos el valor del exploration rate al resolver un problema de aprendizaje reforzado. El problema es que un exploration rate muy bajo y un exploration decay rate muy alto pueden hacer que el agente sea demasiado conservador y pierda oportunidades de descubrimiento y aprendizaje al dedicarse rápidamente en la explotación sin considerar otras opciones que pueden ser mejores, quedando atrapado en soluciones subóptimas. Nuevamente, resurge la importancia de intentar distintos valores para que esos hiperparámetros puedan adecuarse al problema en cuestión.

¿Qué pasa si el exploration decay rate es demasiado bajo?

Si el exploration decay rate es demasiado bajo, entonces la exploración es más prominente y se demora más en converger a una solución, y por ello gaste más recursos computacionales sin necesariamente llegar a una mejor solución.

Por ejemplo, en Flappy bird, un exploration decay rate bajo podría resultar en que se sigan probando saltos aleatorios en lugares y/o inoportunos que no mejoran realmente el rendimiento del modelo, quedándose en un estado de exploración demasiado costoso en tiempo y recursos. A su vez, este tomaría demasiado tiempo en tales tareas pero no en mejorar el resto de su jugada. Por el otro lado, si el exploration decay rate es alto, entonces el modelo profundizaría quizás en unas acciones que inherentemente son subóptimas y nuevamente, al tener una mala base, el juego no tendría el rendimiento esperado.

¿Qué rol cumple la tasa de descuento en Q-Learning ?

La tasa de descuento se encarga de que valorar que tanta "confianza" le tenemos al futuro, es decir, si le damos más valor a la tasa de descuento entonces le damos más valor a las recompensas futuras sobre las presentes. La elección de la tasa de descuento puede afectar significativamente el comportamiento del agente y su capacidad para planificar a largo plazo.

¿Qué tasa de descuento te dio mejores resultados? ¿Por qué crees que fue así?

Dado que una planificar al futuro resulta muy útil ya que en un juego como flappy bird en el que es sumamente importante el estado siguiente al ser crucial para determinar si el pájaro cruza o no la tubería y puede hacer una gigante diferencia si se salta en cierto momento u otro. Se obtuvieron los siguientes resultados al usar un modelo previamente entrenado:
Discount Rate = 0
Game 10 Mean Score 36.3 Record: 81 EXP_RATE: 0.004925504230306071 STEPS: 4160.3
Discount Rate = 0.5
Game 10 Mean Score 52.2 Record: 96 EXP_RATE: 0.004925504230306071 STEPS: 5746.4
Al aumentar la tasa de descuento de 0 a 0.5, en un juego ya entrenado, se aumentó la media 16 puntos, aún más de lo esperado por lo que se confirma que una tasa de descuento mayor puede ser muy beneficiosa al permitir planificar más a futuro.

¿Para qué sirve el learning rate? ¿Qué valor te entregó mejores resultados? Comenta los resultados.
El learning rate tiene que ver con el impacto que causa cada actualización en los valores de la Q-table. Es decir, a mayor learning rate, más grandes son los ajustes al modelo luego de cada iteración, y viceversa.
Es muy importante asegurarse, en todo caso, de que no sea demasiado grande, ya que de lo contrario el modelo podría ser demasiado sensible llevando a divergencias. Observamos lo que ocurre en el experimento:

Learning Rate = 0.3
Game 10 Mean Score 52.2 Record: 96 EXP_RATE: 0.004925504230306071 STEPS: 5746.4
Learning Rate= 0.7
Game 10 Mean Score 30.9 Record: 98 EXP_RATE: 0.004925504230306071 STEPS: 3571.4
Como se puede ver a partir de la experiencia, al menos en un juego ya entrenado, aumentar el learning rate empeoró notablemente el rendimiento, lo que tiene sentido ya que los valores establecidos habían sido ampliamente respaldados mientras que los nuevos podían ser más circunstanciales. De todas formas, quizás al principio del entrenamiento, una learning rate más alta puede ser más beneficiosa ya que no se la q-table inicialmente no tiene información tan valiosa.

2.4 Nueva política de recompensas
Hay varias cosas acerca de la política actual que cambiaría, describiré el porqué de cada una de ellas a continuación. Me parece que la recompensa del agente cuando se encuentra encima o sobre el agujero debiese penalizar de mucho más el saltar de lo que se está penalizando, le establecería -10 mientras que su opuesto +5. Lo anterior ya que considero extremadamente dañino saltar cuando debe bajar el pájaro al no existir una acción que contrarrestre tal salto. En cuanto a su opuesto, es decir, si el agente esta debajo del agujero, entonces saltar y no saltar debiesen tener una equivalencia parecida ya que se puede "contrarrestar" el no saltar (+4 y -4). En cuanto a la pena por perder el juego, me parece que esta es demasiado baja ya que constituye algo muy importante para el juego y el objetivo principal por lo que creo que tendría sentido que contrarrestré a la pena de pasar una tubería al ser de -50 puntos.

Además, en caso de que pierda, agregaría recompensas negativas según que tan lejos quedó del agujero (-1 por unidad de lejanía), ya que no es lo mismo en términos de aprendizaje si casi pasó el agujero a si estuvo muy lejos y un descuento gigante (-100) si es que toca el cielo ya que es totalmente evitable por el modelo y es un fuente de error al principio muy importante.

Bibliografía

- Peralta, D., Herrera-Poyatos, A., & Herrera, F. (2016). Un Estudio sobre el Preprocesamiento para Redes Neuronales Profundas y Aplicación sobre Reconocimiento de Dígitos Manuscritos. Actas de la XVII Conferencia de la Asociación Española para la Inteligencia Artificial, 867-876. Estudio sobre el Preprocesamiento para Redes Neuronales Profundas y Aplicación sobre Reconocimiento de Dígitos Manuscritos.

- Mateo Jiménez, F. (2012). Redes neuronales y preprocesado de variables para modelos y sensores en bioingeniería (Tesis Doctoral). Universidad de Valencia. Redes neuronales y preprocesado de variables para modelos y sensores en bioingeniería.

55 González Molina, F. Aplicación de Redes Neuronales en el cálculo de sobretensiones y tasa de contorneamientos. Aplicación de Redes Neuronales en el cálculo de sobretensiones y tasa de contorneamientos.

- Bushaev, V. (2018, 2 de septiembre). Understanding RMSprop — faster neural network learning. Towards Data Science. https://towardsdatascience.com/understanding-rmsprop-faster-neural-network-learning-62e116fcf29a

- Sotaquirá, M. (2021, 10 de julio). Otros algoritmos de optimización: RMSPROP. En Fundamentos de Deep Learning con Python. Lección 23 de la sección "Redes Neuronales". https://www.codificandobits.com/curso/funda deep-learning-python/redes-neuronales-23-rmsprop/

- Donis-Díaz, Carlos & Morell, Carlos & Valencia Morales, Eduardo. (2009). Comparative study of PSO and back-Propagation for training an MLP neural network in the estimation of the creep rupture stress in ferritic steels. Comparative study of PSO and back-Propagation for training an MLP neural network in the estimation of the creep rupture stress in ferritic steels.

- LearnOpenCV. Understanding Convolutional Neural Network (CNN): A Complete Guide. (S/F). https://learnopencv.com/understanding-convolutional-neural-networks-cnn/

- educative.io, (S/F). What is a neural network flatten layer?. https://www.educative.io/answers/what-is-a-neural-network-flatten-layer

- Makone, A. (2021). Reinforcement Learning 6: Exploration vs Exploitation. https://ashutoshmakone.medium.com/rein learning-5-exploration-vs-exploitation-c1bae5a2ea42