Monitoreo en Proyectos de Machine Learning

Santaigo Garcia - Santiago Loaiza

I. Introducción

Tener un buen monitoreo permite la escalabilidad de un modelo y permite que se puedan detecatar cambios en el modelo evitando que el modelo pierda capacidad para hacer clasificacion, admeas ayuda a identificar en su fase de entrenamiento posibles sesgos o errores del modelo garantizando un modelo transparente y equitativo a la hora de clasificar y tomar decisiones en base a su su proceso de clasificacion

Un problema de que los algoritmos no tengan un monitoreo adecuado son los sesgos, algunos prototipos de IA de NLP tuvieron sesgos bastante notables a tal punto que la misma IA se podria considerar racsita, tambien se han dado caso de IA que su sesgo las lleva a dar respuestas homofobicas, sexistas y machistas.

Si bien las IA a consecuencia de un mal algortimo de aprendizaje pueden llegar a ofender a ciertos sectores, es un problema menor a comparacion de un mal mandejo de Machine learning en ambitos sensibles como son la medicina. En este campo las IA y el Machine Learning han ayudado en los diagnosticos, por lo que en este tipo de herramientas medicas no hay cabida a error en la clasificacion o prediccion de los algoritmos de machine learning.

II. Diseño y entreno del modelo

En otro proyecto se hizo el despliegue de una aplicacion la cual es una herramienta que ayudara a los medicos a la deteccion de un tipo de fractura, para esto se opta por un modelo de capas convolucionales para hacer la clasificaion entre 6 tipos de fractias las cuales son: fractura de mano, dedo, muñeca, antebrazo, humero, codo y hombro.

Para la clasificaion se opta por una red neuronal simple, con tres capas convolucionales, tres capas de max polling, una capa flatten y dos densas, de las cuales en la ultima se especifican la cantidad de clases a clasificar, a continuacion se muestra como es la capa.

```
model = Sequential()

model.add(Conv2D(32,(3,3),1,
          activation="relu",
          input_shape=(128,128,3)))
model.add(MaxPooling2D())
model.add(Conv2D(64,(3,3),1,
          activation="relu"))
model.add(MaxPooling2D())
model.add(Conv2D(32,(3,3),1,
          activation="relu"))
model.add(MaxPooling2D())
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(7,activation='softmax'))
```

El modelo se compila usando *sparse categorical crossentropy* para el tipo de *Loss* y se entrena durante 20 epocas, el final del entrenamiento la validacion para el *Loss* y el *Accuracy* tiene un comportamiento excelente al mostar valores de 0.25 y 0.92, respectivamente .

III. Rendimiento

Una vez completado el entrenamiento se calcula el rendimiento con las metricas de evaluacion principales,*Loss* y el *Accuracy*, esto arroja dos curvas de entrenamiento, la primera es la de *Loss* en la cual tanto para entreno como para validacion va disminuyendo constantemente tal como se ve en la Figura 1.

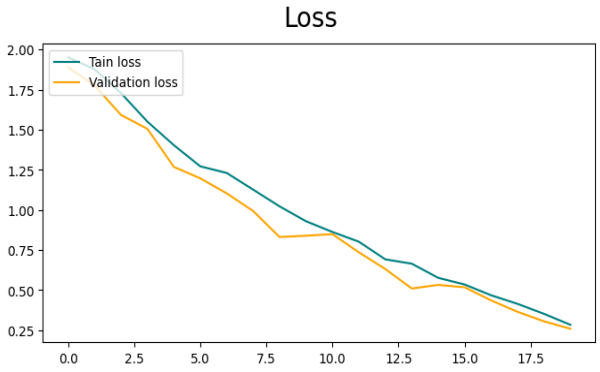

Figura 1. Curva de Perdida

De igual manera se hace la curva del *Accuracy* para el entrenamiento y para la validacion, dando una grafica que aumenta de manera constante.

El comportamiento de estas curvas, tal cual se ve que convergen a un minimo o un maximo, teniendo comportamientos similares entre el entreno y la validacion indican que el entrenamiento del modelo fue satisfactorio

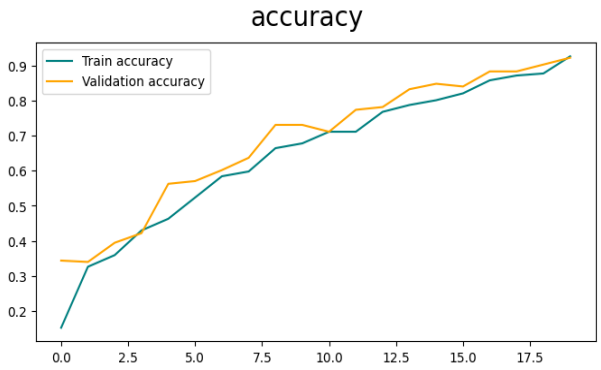

Figura 2. Curva de Exactitud

IV. Monitoreo

Para monitorear el modelo es necesario recurrir a tres medidas principales: El *ReCall*, el cual es la tasa de verdaderos positivos $Recall = TP + FN/TP$, es decir mide la capacidad del modelo para identificar correctamente las instancias positivasen este caso el modelo presenta un *ReCall* de 0.99.

La segunda mediada es la precision definita por $Precision = TP/TP + FP$ el cual es la proporcion de etiquetas identificadas como positivas por el modelo que realmente son positivas, en este caso el modelo tiene una *precision* de 0.99.

Finalmente esta la Exactitud definida por $Accuracy = CorrectPrediction/Totalpredicciones$, es decir la capacidad del modelo para predecir de manera correcta tanto las positivas como las negativas sobre el total de predicciones, el modelo tiene un *Accuracy* de 0.98.

Por otro lado se hace una matriz de confusion para ver como el modelo clasifica de manera correcta los diferentes labels, en este caso se tiene la siguiente matriz de confusion presentada en la Figura 3.

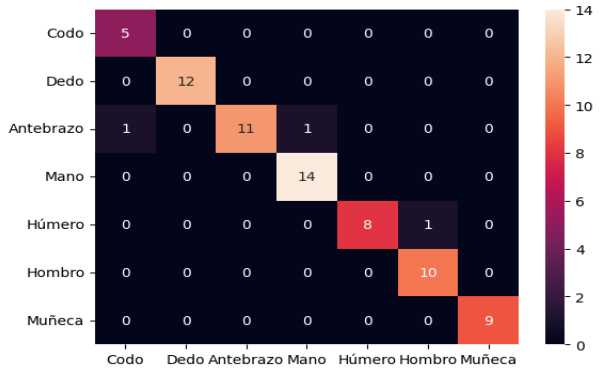

Figura 3. Matriz de confusion

En el caso de la clasificacion de un set de prueba solo hay 3 casos en en los cuales el modelo hacer una prediccion incorrecta, siendo un codo clasificado como antebrazo, un antebrazo clasificado como mano y un hombro calificado como humero. Si bien existen 3 errores en la clasificacion se puede decir que las imagenes mal etiquetadas presentan caracteristicas similares con aquellos lables con las que fueron etiquetadas ya sea porque la fractura es poco apreciable o bien porque las radiograficas de antebrazo, codo, humero y son muy similares ya que varias de las imagenes pertenecientes a este tipo de etiquetas son rediografias tomadas del brazo completo.

V. Conclusiones

Como se observa durante todo el Pipeline del entrenamiento del modelo se puede decir que las redes convolucionales son bastente poderosas para obtener caracteristicas de la imagenes, diferenciarlas y clasificarlas, pues con una arquitectura de una red tan simple que solo tiene tres capas convolucionales so logran obtener resultados bastante buenos.

Es posible mejorar los resultados para obtener un 100 % de precision al tener un conjunto de datos mas amplio o bien un conjunto de datos con el mismo metodo de radriograficas, pues el data set usado muestra imagenes tomadas de diferentes modos, lo que puede llegar a generar ruido dentro de la red neuronal a la hora de hacer clasificaion.

Es importnate aclarar que este tipo de aplicaciones para ser usadas en un ambito profesional deben hacerse en compañia de un profesional de la salud como puede ser un radiologo, quien es el encargado de obntener la data y ayudar a los ingenieros a hacer una etiquetacion correcta de las imagenes, para posteriormente el ingeniero o ingeniera realice el procesamiento de datos, el creacion de la red neuronal, la creacion de la aplicacion y su despliegue.