Experimento 0:

Análisis de *“A Preliminary Study on Deep Transfer Learning Applied to Image Classification for Small Datasets”*

Por Antonio Jesús García Nieto

Índice:

Experimento 0

Introducción

En este experimento pretendemos comprobar si la red de clasificación binaria propuesta en “A Preliminary Study on Deep Transfer Learning Applied to Image Classification for Small Datasets ” [1] se ajusta a nuestro problema de clasificación múltiple y si es mejor entrenar una red mediante scratching o dividiendo el set de datos y aplicando transfer learning.

Precedente

Como hemos comentado el precedente de este experimento es el estudio “A Preliminary Study on Deep Transfer Learning Applied to Image Classification for Small Datasets”.

En el paper los autores defienden el uso de aplicar técnicas de transfer learning sobre técnicas de scratching aún sin estar trabajando con redes pre entrenadas.
Para comprobar su hipótesis inicial los autores trabajan con un dataset de imágenes de células parasitadas o no parasitadas con malaria, es decir, trabajan en una clasificación binaria, por lo que tendremos que realizar algunas adaptaciones al proyecto.

Por otro lado, la red propuesta en el proyecto sigue la arquitectura que se muestra en la siguiente imagen.

Layer (type)	Output shape	Params	Updateable
Conv2D	(None, 48, 48, 32)	896	No
Conv2D	(None, 46, 46, 32)	9,248	No
MaxPooling2D	(None, 23, 23, 32)	0	No
Conv2D	(None, 21, 21, 64)	18,496	No
MaxPooling2D	(None, 10, 10, 64)	0	No
Flatten	(None, 6400)	0	No
Dense	(None, 128)	819,328	Yes
Dense	(None, 2)	258	Yes

Algunos detalles de la implementación son el uso de kernel de tamaño 3x3 para todas la capas de convolución y que las capas de max pooling son de 2x2.

Antes de trabajar con esta arquitectura vamos a destacar su parecido con la red VGG-16[2]. Ciertamente parece una versión reducida de esta familia de redes.

La red SOCO explota una idea vista por primera vez en las redes de la familia VGG, el uso kernels de convolución uniformes de 3x3 apilados.
Como se detalló en la sección de arquitecturas cuando se habló sobre la familia VGG, el usar este tamaño de kernels y apilar capas de convolución genera el mismo efecto que aplicar kernels más grandes y además añade algunas ventajas como ser

computacionalmente más ligeras(siendo C el número de canales, para 3 capas de convolución con un filtro 3x3 tendríamos 3^2*C^2 o lo que es lo mismo 27C^2 parámetros, mientras que para una capa convolucional con un kernel de 7x7 tendríamos 49*C^2 parámetros), además el apilar varias capas causa que se generen capas de rectificación no lineal extras (mientras que con un kernel de 7x7 se genera una, con 3 capas con kernel de 3x3,que es el equivalente, generamos 3).

Una idea que no han tomado de la red VGG y que quizás pudiesen ser beneficiosas es añadir una convolución con un kernel de tamaño 1x1, esto sirve para general no-linealidades extra sin afectar al tamaño del mapa de características.

Metodología

En este estudio queremos tratar dos premisas:

- ¿Podrá la red SOCO servir para nuestro problema de clasificación multiclase?
- ¿Será beneficioso el usar transfer learning en esta red para problemas multiclase?

Para saber si se cumplen las premisas anteriores:
1 - Para comprobar la primera hipótesis, replicamos la red (el código de la implementación se encuentra en el repositorio de github) y adaptamos la red haciendo unas ligeras modificaciones como sustituir la capa de salida (recordemos que el problemas que se aborda en el paper es una clasificación binaria, mientras que nosotros vamos a abordar una clasificación multiclase).

En el estudio,la red propuesta tiene una capa de salida con solo una neurona, con una función de activación sigmoide,lo cual es apropiado para una clasificación binaria, nosotros usaremos una capa de 7 neuronas (una por clase) y una función de activación softmax.

La red recibirá el nombre de CNN_SOCO y la entrenamos con los hiperparametros seleccionados en el estudio:

- Optimizador RMSProp con el lr modificado a 0.0001.
- Batchsize de 128.
- 5 épocas.

2 - Para saber si se cumple la segunda hipótesis realizaremos los mismos experimentos que tienen lugar en el paper.
Para esto, dividiremos el set de datos en subconjuntos y compararemos el resultado con el obtenido en el punto anterior.

Para ello separaremos los datos en dos sets:

- Source: Consiste en la información que se usará para entrenar completamente el modelo. Consiste en un 60% del total del set de datos.
- Target: Consiste en la información que se usará para entrenar las capas de salida del modelo.Se divide a su vez en train(70%) y test(30%)

Experimentos

Para comparar los resultados que obtengamos con los del paper vamos ha hacer los mísmos 4 experimentos que ellos llevan a cabo.

- Caso 1 - CE = train, CT = test
- Caso 2 - CE = source, CT = test
- Caso 3 - CE = source + train, CT = test
- Caso 4 - CE = source, CRE = train, CT = test

Donde CE es conjunto de entrenamiento, CRE conjunto de re.entrenamiento,CT conjunto de test.

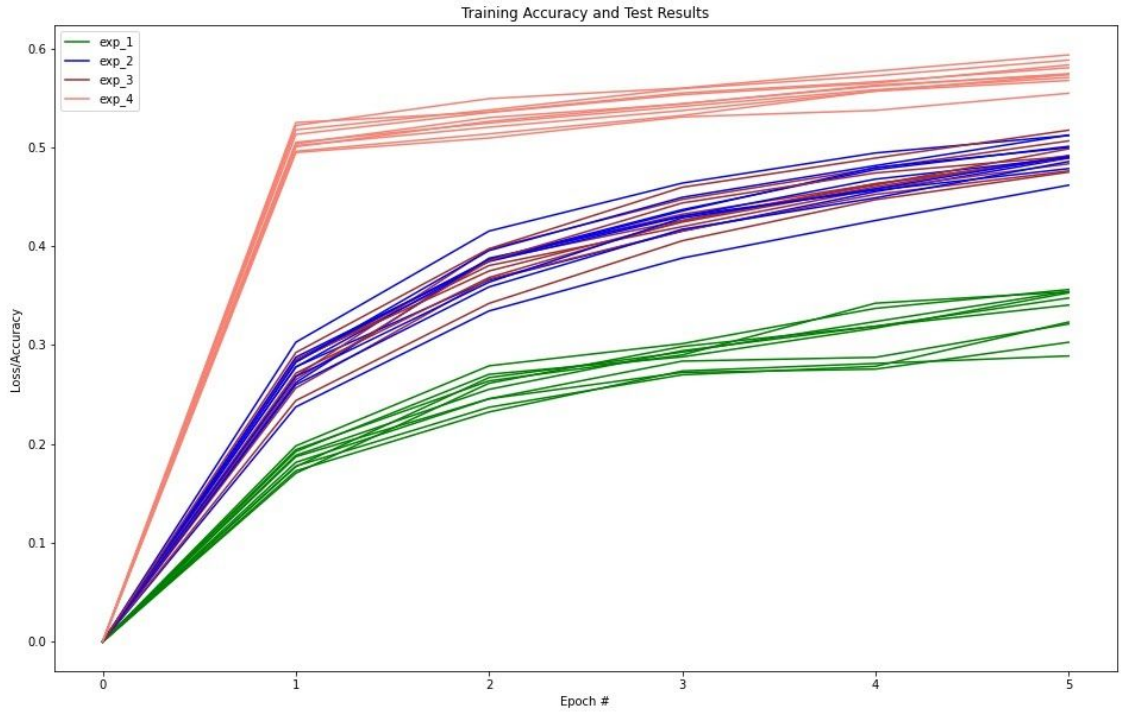
Para demostrar que los resultados son concluyentes se realizarán 10 iteraciones con cada configuración.

Resultados

En la siguiente tabla podemos ver los resultados obtenidos tras testear el modelo.

Iteración	Caso 1	Caso 2	Caso 3	Caso 4
1	0.371633	0.348367	0.402857	0.432449
2	0.365510	0.381837	0.362653	0.446531
3	0.394082	0.464898	0.376939	0.473878
4	0.334694	0.313061	0.370000	0.337959
5	0.336939	0.492653	0.401224	0.428163
6	0.325714	0.313469	0.358775	0.443673
7	0.304082	0.449184	0.283061	0.503878
8	0.319388	0.398367	0.260816	0.350612
9	0.255918	0.327755	0.431633	0.454898
10	0.330408	0.367755	0.264082	0.481020
Media	0.332551	0.374796	0.366327	0.435306

También podemos ver los resultados sobre el set de datos de entrenamiento en forma de gráfico.



Conclusión

Podemos obtener varias conclusiones, si bien es cierto que los resultados son notablemente mejores cuando usamos transfer learning, también podemos darnos cuenta de que los resultados no son demasiado buenos.

La mejora media con los métodos de transfer learning sobre el siguiente mejor resultado es de aproximadamente el 16%. Luego tendremos que recordar esto de cara a un futuro trabajo.

Por otro lado, la precisión en el mejor de los casos es del 0.503878 %. Dado que nuestro objetivo consiste en el diagnóstico de enfermedades, esta precisión no es aceptable. No obstante, basado en otros estudios, podrían añadirse algunas mejoras tanto a nivel arquitectónico (hemos citado alguna en la sección Precedente) como a nivel de hiper-parámetros.

Una forma de mejorar la red modificando solo los hiper-parámetros sería modificar el tamaño del lote, en algunos estudio se pone de relieve que no es buena idea usar tamaños de lote altos y ratios de aprendizaje bajos. Recomendaría probar lotes con un tamaño de 32 [3].

Bibliografía

[1] - Molina M.Á., Asencio-Cortés G., Riquelme J.C., Martínez-Álvarez F. (2021) A Preliminary Study on Deep Transfer Learning Applied to Image Classification for Small Datasets. In: Herrero Á., Cambra C., Urda D., Sedano J., Quintián H., Corchado E. (eds) 15th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2020). SOCO 2020. Advances in Intelligent Systems and Computing, vol 1268. Springer, Cham. https://doi.org/10.1007/978-3-030-57802-2_71

[2] - Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. 689 arXiv:1409.1556 [cs] 2014.

[3] - The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. Ibrahem Kandel, Mauro Castelli. Nova Information Management School (NOVA IMS), Universidade Nova de Lisboa, Campus de Campolide, 1070-312, Lisbon, Portugal.