ModelT Transfer Learning Fine Tuning

With Data Augmentation

Uma imagem com texto, captura de ecrã, Website, Página web

Descrição gerada automaticamente

**Powered By:**

**Gonçalo Ferreira, nº 2222051**

**José Delgado, nº 2222049**

# Introdução

Este documento descreve os resultados obtidos com o modelo que usa técnicas de transfer lerning, fine tunig e data augmentation, reutilizamos várias técnicas que usamos ao longo do projeto, conforme descritas no documento ‘ModelT\_transferLearning\_featureExtraction\_WithoutDataAumentation’, neste documento iremos focar nas funcionalidades novas.

Os passos para realizar o fine tuning a uma rede são os seguintes:

1. Adicionar a nossa rede personalizada em cima de uma rede base já treinada
2. Congelar a rede base (vgg19)
3. Treinar a parte que adicionámos (rede personalizada)
4. Descongelar algumas camadas da rede base (vgg19)
5. Treinar os dois modelos em conjunto o que adicionámos (rede personalizada) e o modelo base(vgg19)

Para acelerar o processo reutilizamos a rede classificadora que já tínhamos treinado no modelo de transfer lerning feature extraction sem data augmentation. Com esta abordagem podemos passar à frente as etapas um, dois e três. Então começamos por importar a VGG19 e de seguida importamos o modelo pré treinado apenas com a parte classificadora. Por fim, bastou apenas ligar os modelos. Primeiro, criamos uma camada de input com o mesmo formato da VGG19. Em seguida, aplicamos uma sequência de data augmentation a esse input. Após a data augmentation, passamos o input processado pelo modelo base VGG19, obtendo a saída do modelo. A saída da VGG19 será o input do modelo pré-treinado. Finalmente, criamos um novo modelo que combina a camada de input, a saída processada pela VGG19 e o modelo pré-treinado.

No notebook existe uma função chamada “print\_layer\_trainable\_status”, que usamos para verificar que camadas estão definidas como treináveis e também para verificar se a união dos modelos (VGG19 + rede pré treinada (classificação)).

Em relação à data augmentation usamos cinco técnicas:

1. Flip Horizontal
   1. Esta técnica inverte a imagem ao longo do eixo horizontal, ajuda o modelo a reconhecer objetos independentemente da orientação (esquerda/direita).
2. Rotação
   1. Aplica uma rotação à imagem dentro de um intervalo específico, ajuda o modelo a aprender independentemente da orientação dos objetos.
3. Zoom
   1. Aplica um zoom, aumentando ou diminuindo o tamanho da imagem, isto ajuda o modelo a aprender objetos de diferentes tamanhos.
4. Contraste
   1. Altera o contraste da imagem, aumentando ou diminuindo a diferença entre as partes claras e escuras da imagem, isto ajuda o modelo a aprender objetos sob diferentes condições de luz e contraste.
5. Ruído Gaussiano
   1. Adiciona ruido gaussiano à imagem, isto ajuda o modelo a ser mais robusto a pequenas variações nos dados, imitando o ruído natural que pode estar presente nas imagens naturalmente em imagens do mundo real.

A seguinte imagem ilustra na prática o uso do ruído Gaussiano

A person wearing a hat

Description automatically generated

Em relação aos dados, dividimos o conjunto completo em três partes distintas, treino, teste e validação. Essa abordagem permitiu-nos treinar o modelo sequencialmente com cada uma das partes do conjunto de treino, avaliando o desempenho em um conjunto de validação separado após cada fase de treino. Posteriormente, os datasets de treino e de validação cada um deles foi dividido em seis partes igualmente proporcionais, garantindo que cada parte representasse aproximadamente um sexto do tamanho original do conjunto, realizamos esta operação com o objetivo de os treinos serem menos demorados, conseguindo obter feedback mais rápido pelo modelo.

Para além das técnicas utilizadas nos modelos anteriores neste modelo usamos também o Model Checkpoint que resumidamente guarda o melhor modelo conseguido ao longo do processo de treino. Este mecanismo é útil para garantir que o melhor desempenho do modelo durante o treino seja preservado e utilizado posteriormente.

Para chegar a este resultado tivemos de realizar vários treinos, no notebook deste modelo estão todos os treinos realizados para chegar aos valores relatados neste documento.

# Resultados

Resultado 1

Primeiramente começamos por descongelar todo o block 5 da VGG19, ou seja, as últimas 5 camadas, isto faz com que os pesos sejam atualizados ao longo do processo de treino, e realizamos alguns treinos.

A seguinte figura ilustra uma representação da VGG19 com o block 5 definido como treinável. Relembrar que a camada de max pooling do block 5 não possui pesos treináveis

A diagram of a diagram of a diagram

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generatedA seguinte figura mostra as técnicas que usamos para realizar os treinos e os parâmetros que utilizamos para realizar o primeiro treino, nela conseguimos ver a nova técnica usada, o Model Checkpoint.

A seguinte figura mostra a realização do primeiro treino com o subdataset 1, no treino foram realizadas dez épocas e conseguimos ver que com o primeiro treino o modelo consegui logo chegar a uma validation accuracy de 87%, também é possível ver a importância do Model Checkpoint, pois no final da última época o modelo que terminou não foi o melhor, tendo sido guardado o melhor modelo no final da oitava época.

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generatedA seguinte figura mostra o último treino realizado com o block 5 da VGG19 congelado, na figura observamos que o modelo desde o primeiro treino até o último não teve melhorias, felizmente não obteve overfitting.

Devido ao resultado descrito acima iremos descongelar mais camadas da VGG19 e retirar novas conclusões sobre os resultados obtidos. Se forem melhores continuamos a descongelar camadas se não paramos e o modelo fica finalizado.

Resultado 2

Como referido no resultado 1 iriamos descongelar mais camadas e está parte demostrará os resultados obtidos.

Primeiramente começamos por descongelar todo o block 4, ou seja, agora as últimas 10 da VGG19 camadas ficaram definidas como treináveis.

A seguinte figura ilustra uma representação da VGG19 com o block 5 e o block 4 definidos como treináveis.

A diagram of a diagram of a diagram

Description automatically generated

Com o block 4 e o block 5 descongelados realizamos sete treinos, ao longo dos treinos o calback ReduceLROnPlateau reduzia cada vez mais o lerning rate e fomos à medida que os treinos avançavam usando o lerning rate que era sugerido, iniciamos estes treinos com um lerning rate de 1e-2 e acabamos por terminar o setimo treino com um lerning rate de 1e-12, o que demostra a importancia da utilizacao deste callback.

A screenshot of a computer code

Description automatically generatedA seguinte figura mostra o resultado do último treino que fizemos com as camadas do block 4 e block 5 defenidos como treináveis. Conseguimos observar que os resultados são bastante positivos, chegando agora a uma validation accuracy de 90%.

A seguinte figura mostra o gráfico que ilustra os resultados do modelo, nele conseguimos ver que o modelo não tem overfitting e que a validation accuracy se encontra nos 90%, iremos descongelar o block 3 da VGG19 para tentar obter melhores resultados.

A graph of a line

Description automatically generated with medium confidence

Resultado Final

Fomos ainda descongelar o block 3 e após alguns treinos verificamos que o modelo não conseguiu obter melhores resultados, na verdade até começou a dar sinais de overfitting e a accuracy da validação mantendo-se nos 90%. Portanto devido ao uso do calback Model Checkpoint o modelo final ficou guardado e foi atingido em treinos anteriores com apenas o block 4 e block 5 descongelados.

A seguinte figura ilustra uma representação da VGG19 com o block 5, block 4 e o block 3 definidos como treináveis, ou seja, as últimas 15 camadas ajustaram os seus pesos ao longo do processo de treino.

A diagram of a diagram of a diagram

Description automatically generated

As seguintes figuras mostram os resultados ao descongelar as camadas do block 3, nela conseguimos verificar o que foi relatado anteriormente, ou seja, o modelo não obteve melhores resultados, começando a dar evidencias de overfitting.

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a graph

Description automatically generated

A figura abaixo ilustra o resultado final com o dataset de teste, conseguimos observar que a validation accuracy está nos 89%.

A close-up of a computer screen

Description automatically generated

O modelo anterior igual a este, mas sem data augmentation teve resultados bastante parecidos. Depois de algumas pesquisas podemos ter algumas respostas para este acontecimento, como, a arquitetura escolhida pois ambos usavam a mesma, a complexidade do problema, a qualidade do dataset e as técnicas utilizadas na data augmentation. Podem existir inúmeras razoes, tentamos perceber quais delas poderão ter acontecido.