Maior Dúvida da Aula

MC886/MO444, November 3, 2022

1. Não entendi muito bem o princípio do skip connection, por que ele funciona melhor na ResNet do que os outros modelos? Não entendi o ganho que traz, por que ele deixa a análise melhor?

2. Na rede ResNet, é feita uma soma aritmética (element-wise) ou uma concatenação de entradas com mapas de ativação? Como é feita essa soma/concatenação?

3. Não entendi porque a camada de 'bottleneck' não é utilizada em arquiteturas menores. Existe alguma perda de informação referente a essa operação que faça com que ela só seja interessante em redes mais profundas?

4. Por que é comum ouvir “Se uma ResNet não resolver seu problema, o problema está ...” ?

5. Sobre as inceptions module. Tenho uma entrada 28x28x192 e aplico um filtro ~~1x1x16~~ 16 filtros 1x1x192 convolucionais nessa entrada com o intuito de diminuir a dimensionalidade para a aplicação do próximo filtro, certo? Existe uma relação entre o tamanho da profundidade da entrada com meu filtro 1x1? Porque utilizar um 1x1x16 e não um 1x1x32? Seria porque eu estaria perdendo mais informações?

6. Durante a explicação dos filtros 1x1, foi comentado que a saída do MaxPooling manteria a dimensão do feature map de entrada. Não sei se entendo o MaxPooling corretamente, mas ele não diminuiria o tamanho da feature map em 1/4 (W/2 x Hx2)?

7. Podemos usar batch normalization em camadas de convolução? Como funciona a normalização nesse caso?

8. CNNs são boas para o processamento de texto?

9. É possível utilizar convolução para dados tabulares? Eu vejo muitos exemplos apenas com imagens.

10. Quanto tempo (em média) para treinar uma rede CNN “do zero”?

11. Com essas estruturas de deep learning mais complexas como o GoogleNet, percebo que a questão de custo sempre entra em pauta. Quando você fala custo e “caro”, é caro em dinheiro? Ou caro em tempo de processamento? Se for em dinheiro, é caro quanto?

Transfer Learning

Machine Learning

Prof. Sandra Avila
Institute of Computing (IC/Unicamp)

MC886/MO444, November 3, 2022

Today's Agenda

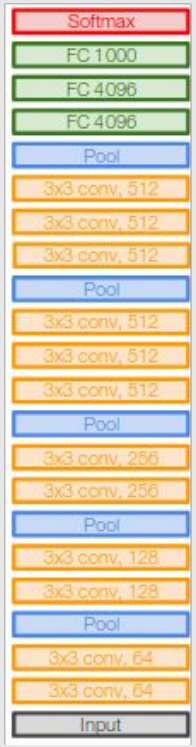
- Transfer Learning
  - What?
  - How?
  - When?

Transfer Learning

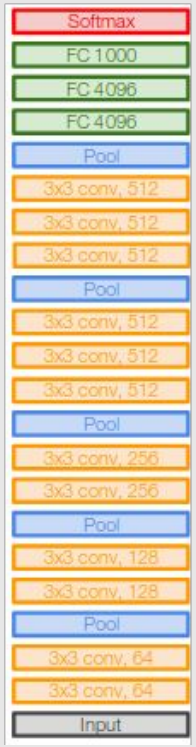
Motivation for transfer learning is based on the fact that **people can intelligently apply knowledge learned previously for a different task** or domain that can be used **to solve new problems faster** or with better solutions.

Transfer Learning

Transfer Learning with CNNs

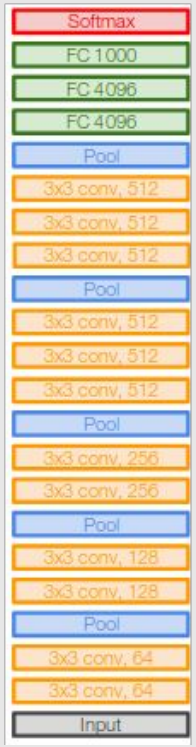


Transfer Learning with CNNs

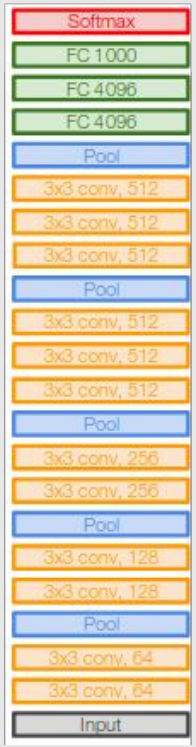


Train on ImageNet
(or large dataset)

Transfer Learning with CNNs

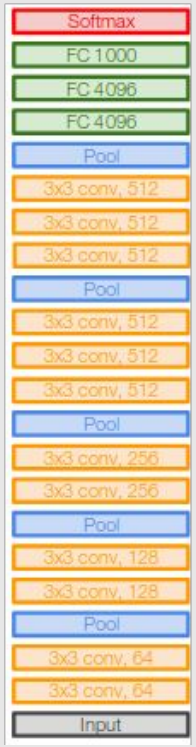


Train on ImageNet

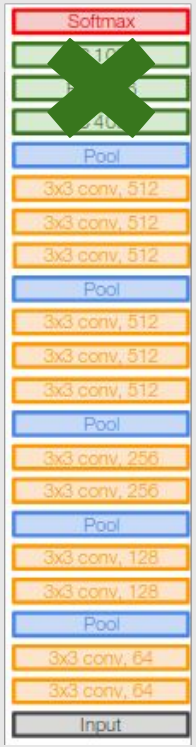


Small dataset (C classes):
Transfer learning with fine-tuning

Transfer Learning with CNNs

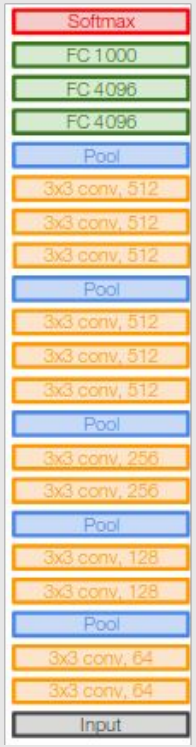


Train on ImageNet

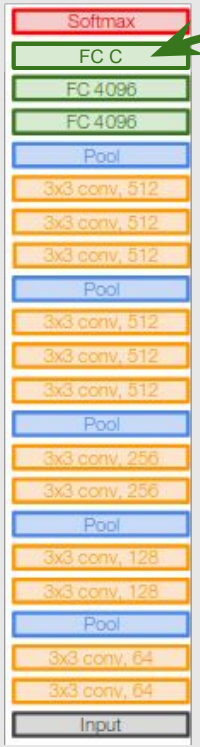


Small dataset (C classes):
Transfer learning with fine-tuning

Transfer Learning with CNNs

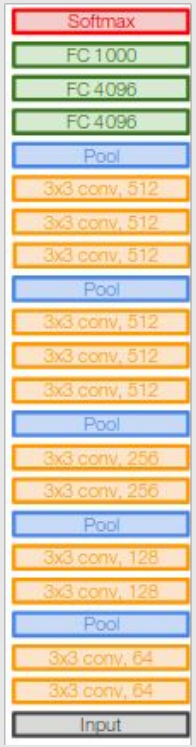


Train on ImageNet

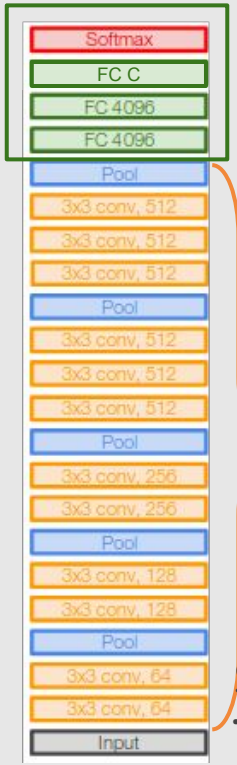


Small dataset (C classes):
Transfer learning with fine-tuning

Transfer Learning with CNNs



Train on ImageNet



Reinitialize this and train

Freeze these

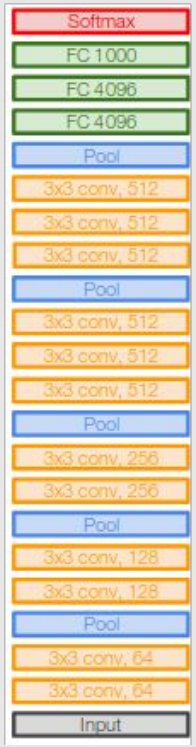
Small dataset (C classes):
Transfer learning with fine-tuning

```
# Carregamento do modelo pré-treinado SEM as camadas densas
# (include_top = False)
model = tf.keras.applications.ResNet50(weights='imagenet', include_top=False)

# Congela camadas pré-treinadas
for layer in model.layers:
    layer.trainable = False

# Insere novas camadas no fim da rede para classificação
frozen_model = tf.keras.Sequential([
    model,
    tf.keras.layers.GlobalAveragePooling2D(),
    tf.keras.layers.Dense(10, activation='softmax')
])
```

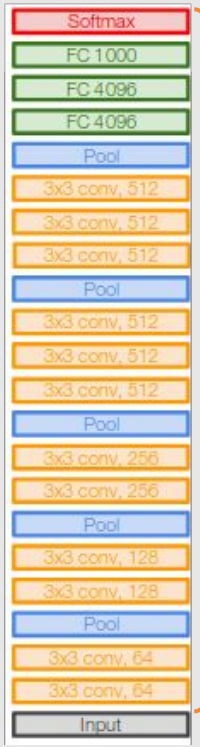
```
# Carregamento do modelo pré-treinado SEM as camadas densas
# (include_top
model = tf.ker                                                  le_top=False)

# Congela cama
for layer in m
    layer.trai

# Insere novas
frozen_model =
    model,
    tf.keras.l
    tf.keras.l
])
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 1, 1, 2048)        23587712

 global_average_pooling2d (G  (None, 2048)             0
 lobalAveragePooling2D)

 dense (Dense)               (None, 10)                20490

=================================================================
Total params: 23,608,202
Trainable params: 20,490
Non-trainable params: 23,587,712
_________________________________________________________________
```

Transfer Learning with CNNs



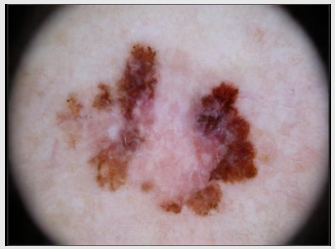
Train on ImageNet

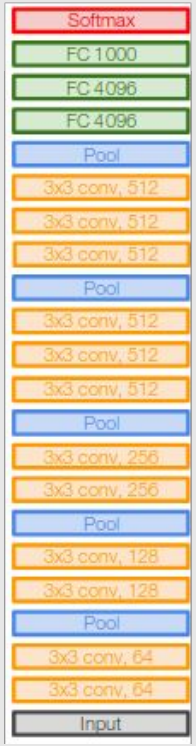


Freeze these

Small dataset (C classes):
Transfer learning without fine-tuning

Transfer Learning with CNNs

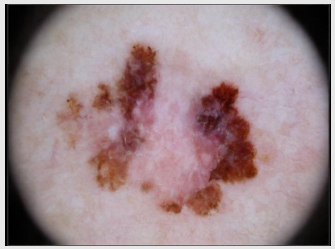


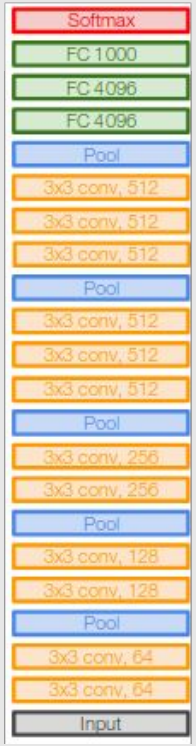


[0.01 0.8 1 0.5 … 0.3 0.07 0 0.4 0.6 0 0]
4096-d

VGG as **Feature Extractor**

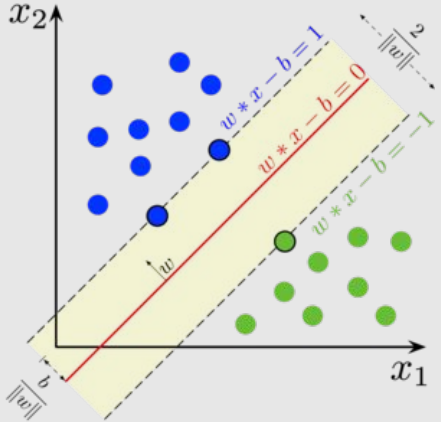
Transfer Learning with CNNs





[0.01 0.8 1 0.5 … 0.3 0.07 0 0.4 0.6 0 0]
4096-d

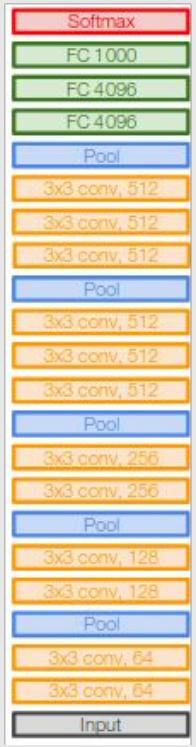
Train a classifier (e.g., SVM)



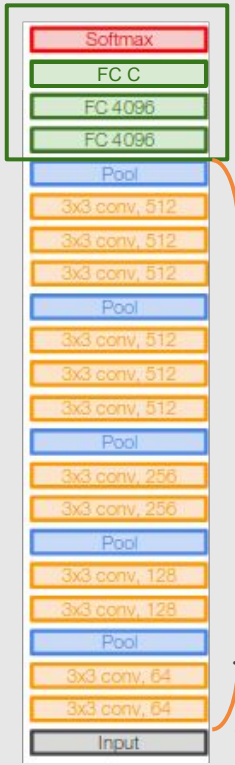
VGG as **Feature Extractor**

```
# Carregamento do modelo pré-treinado SEM as camadas densas
# (include_top = False)
model = tf.keras.applications.ResNet50(weights='imagenet', include_top=False)

# Congela camadas pré-treinadas
for layer in model.layers:
    layer.trainable = False

frozen_model = tf.keras.Sequential([
    model,
    tf.keras.layers.GlobalAveragePooling2D()
])

feats_train = frozen_model.predict(X_train)
feats_val = frozen_model.predict(X_val)
```

Transfer Learning with CNNs



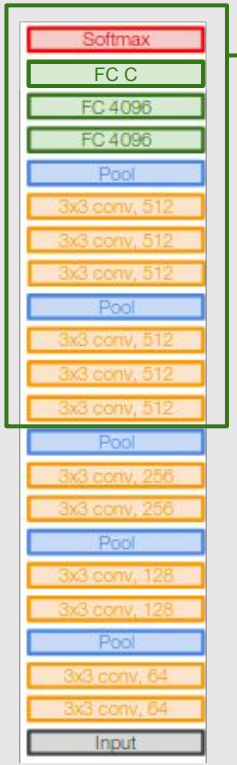
Train on ImageNet



Reinitialize this and train

Freeze these

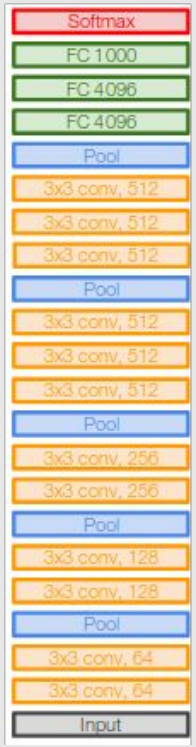
Transfer learning with fine-tuning



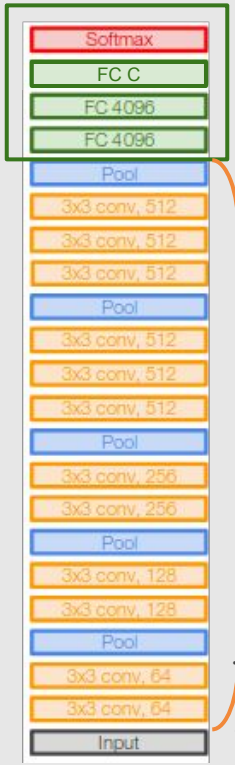
Train this

Medium dataset (C classes):
Transfer learning with fine-tuning

Transfer Learning with CNNs



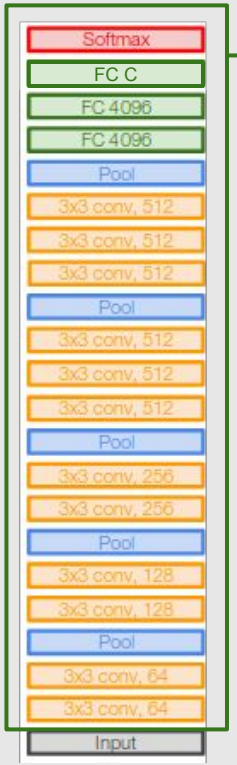
Train on ImageNet



Reinitialize this and train

Freeze these

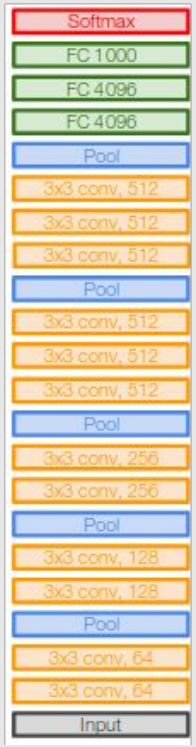
Transfer learning with fine-tuning



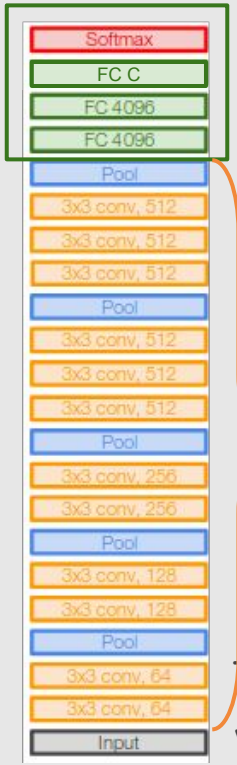
Train this

Medium dataset (C classes): **Transfer learning with fine-tuning**

Transfer Learning with CNNs



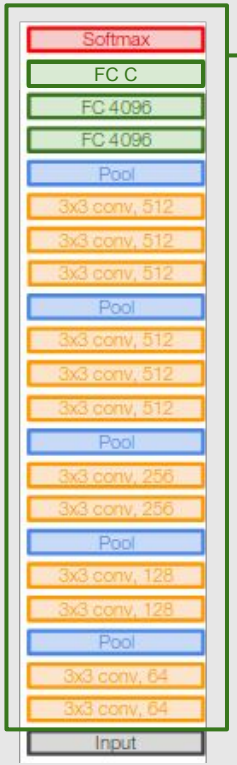
Train on ImageNet



Reinitialize this and train

Freeze these

Transfer learning with fine-tuning



Train this

Lower learning rate when fine-tuning; 1/10 of original LR is good starting point

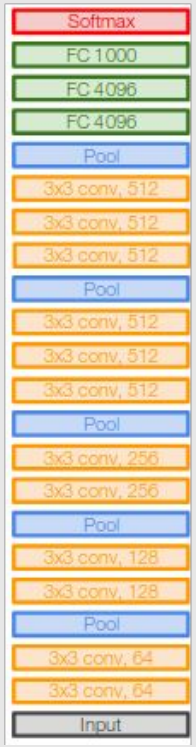
Transfer learning with fine-tuning

```python
# Carregamento do modelo pré-treinado SEM as camadas densas
# (include_top = False)
model = tf.keras.applications.ResNet50(weights='imagenet', include_top=False)

# Descongela camadas pré-treinadas
for layer in model.layers:
    layer.trainable = True

# Insere novas camadas no fim da rede para classificação
frozen_model = tf.keras.Sequential([
    model,
    tf.keras.layers.GlobalAveragePooling2D(),
    tf.keras.layers.Dense(10, activation='softmax')
])
```

```
# Carregamento do modelo pré-treinado SEM as camadas densas
# (include_top = False)
model = tf.keras.applications.ResNet50(weights='imagenet', include_top=False)

# Descongela c
for layer in m
    layer.trai

# Insere novas
frozen_model =
    model,
    tf.keras.l
    tf.keras.l
])
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 1, 1, 2048)        23587712

 global_average_pooling2d_2   (None, 2048)             0
 (GlobalAveragePooling2D)

 dense_1 (Dense)             (None, 10)                20490

=================================================================
Total params: 23,608,202
Trainable params: 23,555,082
Non-trainable params: 53,120
_________________________________________________________________
```

```
# Carregamento do modelo pré-treinado SEM as camadas densas
# (include_top = False)
model = tf.keras.applications.ResNet50(weights='imagenet', include_top=False)

# Descongela camadas pré-treinadas
for layer in model.layers[:8]:
    layer.trainable = False
for layer in model.layers[8:]:
    layer.trainable = True

# Insere novas camadas no fim da rede para classificação
frozen_model = tf.keras.Sequential([
    model,
    tf.keras.layers.GlobalAveragePooling2D(),
    tf.keras.layers.Dense(10, activation='softmax')
])
```



More specific

More generic

	Very similar dataset	Very different dataset
Very little data	?	?
Quite a lot of data	?	?

https://distill.pub/2017/feature-visualization

Edges | Textures | Patterns | Parts | Objects

More generic ⟷ **More specific**

https://distill.pub/2017/feature-visualization

To be continued ...