Índice

[Material 2](#_Toc186732731)

[ResNet 2](#_Toc186732732)

[Tentativa 1 2](#_Toc186732733)

[Tentativa 2 3](#_Toc186732734)

[Material + Ano 5](#_Toc186732735)

[ResNet 5](#_Toc186732736)

[Tentativa 1 5](#_Toc186732737)

[Do Zero 5](#_Toc186732738)

[Com Tentativa 1 – Material 6](#_Toc186732739)

[Tentativa 2 7](#_Toc186732740)

[Do Zero 7](#_Toc186732741)

[Com Tentativa 2 – Material 8](#_Toc186732742)

[Material + Ano + Estrutura 9](#_Toc186732743)

# Material

Neste conjunto de códigos, foi apenas utilizado a propriedade Material para treinar o modelo. Existem 3 classes para esta propriedade e utilizei a pasta "Pasta Final TFRecord – Material" nos modelos, uma vez que esta contém os ficheiros TFRecord. Abaixo, pode-se observar como os dados estão separados (treino, validação e teste) para cada classe. Esta tabela encontra-se presente no ficheiro "Classes.xlsx".

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Classe** | **Material** | **Quantidade** | **Treino** | **Validação** | **Teste** |
| 0 | Alvenaria | 186 | 146 | 23 | 17 |
| 1 | Aço | 143 | 114 | 13 | 16 |
| 2 | Betão Armado | 6 130 | 4 907 | 609 | 614 |
|  | Total | 6 459 | 5 167 | 645 | 647 |

## ResNet

### Tentativa 1

Caminho no GitHub para o código: Material -> ResNet -> Tentativa 1 -> Material (ResNet) – Tentativa 1.py

Caminho no GitHub para os resultados: Material -> ResNet -> Tentativa 1 e de forma resumida Resultados -> Material -> ResNet -> Tentativa 1.xlsx

**Website utilizado:** <https://keras.io/guides/transfer_learning/>

**O modelo está pré-treinado?** Sim, o modelo ResNet50 é pré-treinado com pesos do ImageNet.

**Tem otimização de hiper parâmetros?** Não, não há otimização de hiperparâmetros implementada.

**Qual é o número de epochs?** 10 epochs.

**Tem regra de paragem antecipada?** Não, não há regra de paragem antecipada.

O resumo do código é o seguinte:

* É carregado o modelo ResNet50 sem a camada final de classificação (include\_top=False), utilizando pesos do ImageNet.
* As camadas do modelo ResNet50 são congeladas para manter as características pré-treino.
* Um novo modelo é construído com entrada de tamanho (224x224x3) e saída composta por uma camada densa com 3 classes e ativação softmax.
* O modelo é compilado com o otimizador Adam, a função de perda sparse\_categorical\_crossentropy e métrica accuracy.
* O conjunto de dados é carregado a partir de ficheiros TFRecord Material, onde as imagens são redimensionadas e associadas às suas labels.
* O modelo é treinado durante 10 épocas com dados de treino e validação.
* É avaliado num conjunto de teste, apresentando os valores finais de perda (loss) e exatidão (accuracy).

### Tentativa 2

Caminho no GitHub para o código: Material -> ResNet -> Tentativa 2 -> Material (ResNet) – Tentativa 2.py

Caminho no GitHub para os resultados: Material -> ResNet -> Tentativa 2 e de forma resumida Resultados -> Material -> ResNet -> Tentativa 2.xlsx

**Website utilizado:** <https://keras.io/guides/transfer_learning/>

**O modelo está pré-treinado?** Sim, o modelo ResNet50 é pré-treinado com pesos do ImageNet.

**Tem otimização de hiper parâmetros?** Não, não há otimização de hiperparâmetros implementada.

**Qual é o número de epochs?** 10 epochs.

**Tem regra de paragem antecipada?** Não, não há regra de paragem antecipada.

O resumo do código é o seguinte:

* É carregado o modelo ResNet50 sem a camada final de classificação (include\_top=False), utilizando pesos do ImageNet.
* As camadas do modelo ResNet50 são congeladas para manter as características pré-treino.
* Um novo modelo é construído com entrada de tamanho (224x224x3) e saída composta por uma camada densa com 3 classes e ativação softmax.
* Algumas camadas do modelo ResNet50 são descongeladas para fine-tuning (As 10 últimas).
* O modelo é compilado com o otimizador Adam, a função de perda sparse\_categorical\_crossentropy e métrica accuracy.
* O conjunto de dados é carregado a partir de ficheiros TFRecord Material, onde as imagens são redimensionadas e associadas às suas labels.
* O modelo é treinado durante 10 épocas com dados de treino e validação.
* É avaliado num conjunto de teste, apresentando os valores finais de perda (loss) e exatidão (accuracy).

# Material + Ano

Neste conjunto de códigos, foi apenas utilizado a propriedade Material e Ano para treinar o modelo. Existem 5 classes para esta propriedade e utilizei a pasta "Pasta Final TFRecord – Material + Ano" nos modelos, uma vez que esta contém os ficheiros TFRecord. Abaixo, pode-se observar como os dados estão separados (treino, validação e teste) para cada classe. Esta tabela encontra-se presente no ficheiro "Classes.xlsx".

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Classe** | **Material** | **Ano** | **Qtd** | **Treino** | **Validação** | **Teste** |
| 0 | Alvenaria |  | 186 | 155 | 19 | 12 |
| 1 | Aço | Antes de 1983 | 64 | 54 | 5 | 5 |
| 2 | Aço | Depois de 1983 | 79 | 62 | 7 | 10 |
| 3 | Betão Armado | Antes de 1983 | 579 | 464 | 58 | 57 |
| 4 | Betão Armado | Depois de 1983 | 5 551 | 4 432 | 556 | 563 |
|  |  | Total | 6459 | 5167 | 645 | 647 |

## ResNet

### Tentativa 1

#### Do Zero

Caminho no GitHub para o código: Material + Ano -> ResNet -> Tentativa 1 -> Do Zero -> Material + Ano (ResNet) - Tentativa 1 Do Zero.py

Caminho no GitHub para os resultados: Material + Ano -> ResNet -> Tentativa 1 -> Do Zero e de forma resumida Material + Ano -> ResNet -> Tentativa 1 -> Do Zero.xlsx

**Website utilizado:** <https://keras.io/guides/transfer_learning/>

**O modelo está pré-treinado?** Sim, o modelo ResNet50 é pré-treinado com pesos do ImageNet.

**Tem otimização de hiper parâmetros?** Não, não há otimização de hiperparâmetros implementada.

**Qual é o número de epochs?** 10 epochs.

**Tem regra de paragem antecipada?** Não, não há regra de paragem antecipada.

O resumo do código é o seguinte:

* É carregado o modelo ResNet50 sem a camada final de classificação (include\_top=False), utilizando pesos do ImageNet.
* As camadas do modelo ResNet50 são congeladas para manter as características pré-treino.
* Um novo modelo é construído com entrada de tamanho (224x224x3) e saída composta por uma camada densa com 5 classes e ativação softmax.
* O modelo é compilado com o otimizador Adam, a função de perda sparse\_categorical\_crossentropy e métrica accuracy.
* O conjunto de dados é carregado a partir de ficheiros TFRecord Material, onde as imagens são redimensionadas e associadas às suas labels.
* O modelo é treinado durante 10 épocas com dados de treino e validação.
* É avaliado num conjunto de teste, apresentando os valores finais de perda (loss) e exatidão (accuracy).

#### Com Tentativa 1 – Material

Caminho no GitHub para o código: Material + Ano -> ResNet -> Tentativa 1 -> Com Tentativa 1 – Material-> Material + Ano (ResNet) - Tentativa 1 Com Tentativa 1 – Material.py

Caminho no GitHub para os resultados: Material + Ano -> ResNet -> Tentativa 1 -> Com Tentativa 1 – Material e de forma resumida Material + Ano -> ResNet -> Tentativa 1 -> Com Tentativa 1 – Material.xlsx

**Website utilizado:** <https://keras.io/guides/transfer_learning/>

**O modelo está pré-treinado?** Sim, o modelo ResNet50 é pré-treinado com pesos do ImageNet.

**Tem otimização de hiper parâmetros?** Não, não há otimização de hiperparâmetros implementada.

**Qual é o número de epochs?** 10 epochs.

**Tem regra de paragem antecipada?** Não, não há regra de paragem antecipada.

O resumo do código é o seguinte:

* É carregado o modelo a partir do ficheiro Tentativa 1 – Material.
* A última camada do modelo base é removida para adaptar a saída para um novo modelo.
* As camadas do modelo ResNet50 são congeladas para manter as características pré-treino.
* Um novo modelo é construído com entrada de tamanho (224x224x3) e saída composta por uma camada densa com 5 classes e ativação softmax.
* O modelo é compilado com o otimizador Adam, a função de perda sparse\_categorical\_crossentropy e métrica accuracy.
* O conjunto de dados é carregado a partir de ficheiros TFRecord Material + Ano, onde as imagens são redimensionadas e associadas às suas labels.
* O modelo é treinado durante 10 épocas com dados de treino e validação.
* É avaliado num conjunto de teste, apresentando os valores finais de perda (loss) e exatidão (accuracy).

### Tentativa 2

#### Do Zero

Caminho no GitHub para o código: Material + Ano -> ResNet -> Tentativa 2 -> Do Zero -> Material + Ano (ResNet) - Tentativa 2 Do Zero.py

Caminho no GitHub para os resultados: Material + Ano -> ResNet -> Tentativa 2 -> Do Zero e de forma resumida Material + Ano -> ResNet -> Tentativa 2 -> Do Zero.xlsx

**Website utilizado:** <https://keras.io/guides/transfer_learning/>

**O modelo está pré-treinado?** Sim, o modelo ResNet50 é pré-treinado com pesos do ImageNet.

**Tem otimização de hiper parâmetros?** Não, não há otimização de hiperparâmetros implementada.

**Qual é o número de epochs?** 10 epochs.

**Tem regra de paragem antecipada?** Não, não há regra de paragem antecipada.

O resumo do código é o seguinte:

* É carregado o modelo ResNet50 sem a camada final de classificação (include\_top=False), utilizando pesos do ImageNet.
* As camadas do modelo ResNet50 são congeladas para manter as características pré-treino.
* Um novo modelo é construído com entrada de tamanho (224x224x3) e saída composta por uma camada densa com 5 classes e ativação softmax.
* Algumas camadas do modelo ResNet50 são descongeladas para fine-tuning (As 2 últimas).
* O modelo é compilado com o otimizador Adam, a função de perda sparse\_categorical\_crossentropy e métrica accuracy.
* O conjunto de dados é carregado a partir de ficheiros TFRecord Material + Ano, onde as imagens são redimensionadas e associadas às suas labels.
* O modelo é treinado durante 10 épocas com dados de treino e validação.
* É avaliado num conjunto de teste, apresentando os valores finais de perda (loss) e exatidão (accuracy).

#### Com Tentativa 2 – Material

Caminho no GitHub para o código: Material + Ano -> ResNet -> Tentativa 2 -> Com Tentativa 2 – Material-> Material + Ano (ResNet) - Tentativa 2 Com Tentativa 2 – Material.py

Caminho no GitHub para os resultados: Material + Ano -> ResNet -> Tentativa 2 -> Com Tentativa 2 – Material e de forma resumida Material + Ano -> ResNet -> Tentativa 2 -> Com Tentativa 2 – Material.xlsx

**Website utilizado:** <https://www.tensorflow.org/api_docs/python/tf/keras/models/load_model>; [https://stackoverflow.com/questions/55392591/how-to-remove-the-last-layer-from-trained-model-in-tensorflow?](https://stackoverflow.com/questions/55392591/how-to-remove-the-last-layer-from-trained-model-in-tensorflow?utm_source=chatgpt.com); <https://github.com/thmstrmnd/transferlearning-resnet50-model/blob/master/transferlearning-resnet50-model.ipynb>

**O modelo está pré-treinado?** Sim, o modelo base é carregado do ficheiro Tentativa 2 – Material.

**Tem otimização de hiper parâmetros?** Não, não há otimização de hiperparâmetros implementada.

**Qual é o número de epochs?** 10 epochs.

**Tem regra de paragem antecipada?** Não, não há regra de paragem antecipada.

O resumo do código é o seguinte:

* É carregado o modelo a partir do ficheiro Tentativa 2 – Material.
* A última camada do modelo base é removida para adaptar a saída para um novo modelo.
* As camadas do modelo ResNet50 são congeladas para manter as características pré-treino.
* Um novo modelo é construído com entrada de tamanho (224x224x3) e saída composta por uma camada densa com 5 classes e ativação softmax.
* Algumas camadas do modelo ResNet50 são descongeladas para fine-tuning (Última).
* O modelo é compilado com o otimizador Adam, a função de perda sparse\_categorical\_crossentropy e métrica accuracy.
* O conjunto de dados é carregado a partir de ficheiros TFRecord Material + Ano, onde as imagens são redimensionadas e associadas às suas labels.
* O modelo é treinado durante 10 épocas com dados de treino e validação.
* É avaliado num conjunto de teste, apresentando os valores finais de perda (loss) e exatidão (accuracy).

# Material + Ano + Estrutura