Laboratório 6 - Treinar um modelo neural de tradução

Diário de bordo

Código:
https://colab.research.google.com/drive/13YMtt4IrbYL9T3SoVbXEG0MrqHiPnRBx?usp=sharing

1 **Hyperparâmetros:**

source_embedding_size=24

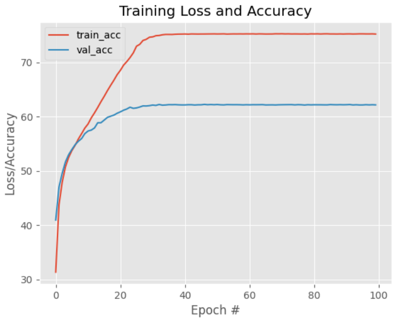
target_embedding_size=24

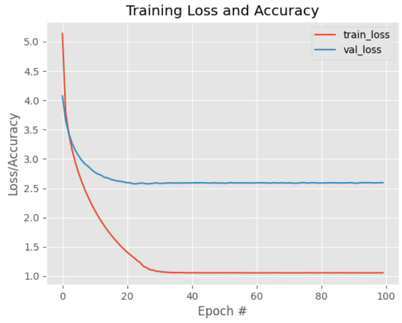
encoding_size=32

batch_size=32

num_epochs=100

learning_rate=0.0005





```
(encoder): NMTEncoder(
  (source_embedding): Embedding(3025, 24, padding_idx=0)
  (birnn): GRU(24, 32, batch_first=True, bidirectional=True)
(decoder): NMTDecoder(
  (target_embedding): Embedding(4911, 24, padding_idx=0)
  (gru_cell): GRUCell(88, 64)
  (hidden_map): Linear(in_features=64, out_features=64, bias=True)
  (classifier): Linear(in_features=128, out_features=4911, bias=True)
```

```
INFO] epoch 22, train loss 1.3188232963545286, val loss 2.5745678397475698
[INFO] train_acc 70.17609968704853, val acc 61.43141399014763
[INFO] best validation loss updated and checkpoint saved
```

Evaluate NMT

```
[INFO] mean result: 0.4276705749353432, median result: 0.4061551592130281
```

Comentários:

Overfitting, vamos diminuir a taxa de aprendizagem.

2 **Hyperparâmetros:**

source_embedding_size=24

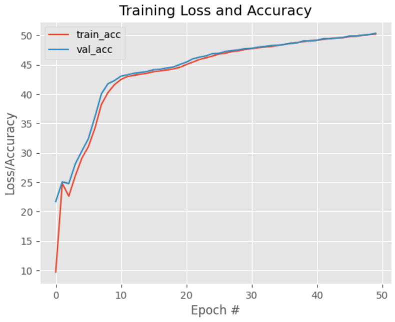
target_embedding_size=24

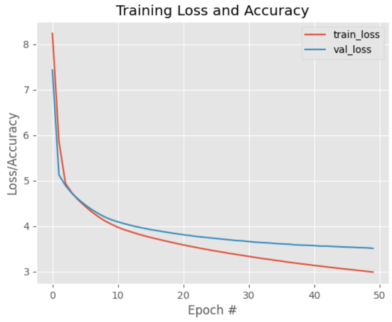
encoding_size=32

batch_size=32

num_epochs=**50**

learning_rate=**0.00005**





```
[INFO] using device cuda
NMTModel(
  (encoder): NMTEncoder(
    (source_embedding): Embedding(3025, 24, padding_idx=0)
    (birnn): GRU(24, 32, batch_first=True, bidirectional=True)
  )
  (decoder): NMTDecoder(
    (target_embedding): Embedding(4911, 24, padding_idx=0)
    (gru_cell): GRUCell(88, 64)
    (hidden_map): Linear(in_features=64, out_features=64, bias=True)
    (classifier): Linear(in_features=128, out_features=4911, bias=True)
  )
)
```

```
[INFO] epoch 49, train loss 2.9866147559985787, val loss 3.5085229209212003
[INFO] train_acc 50.221524468939506, val acc 50.350192252949405
[INFO] best validation loss updated and checkpoint saved
```

Evaluate NMT

```
[INFO] mean result: 0.28632547839254285, median result: 0.2703810696954543
```

Comentários:

Os resultado pioraram em relação ao 1º treinamento.

3 **Hyperparâmetros:**

source_embedding_size=**64**

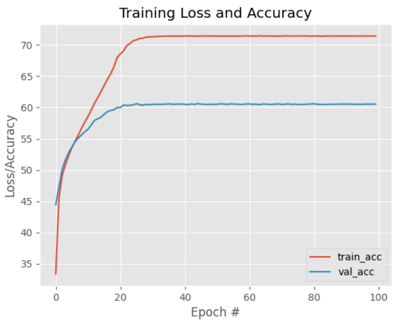
target_embedding_size=**64**

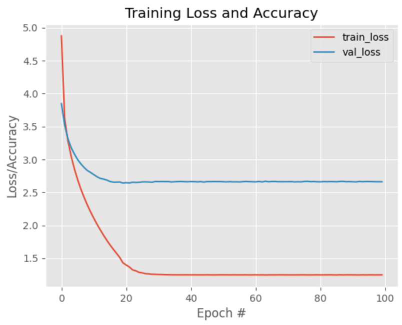
encoding_size=32

batch_size=32

num_epochs=**100**

learning_rate=**0.0005**





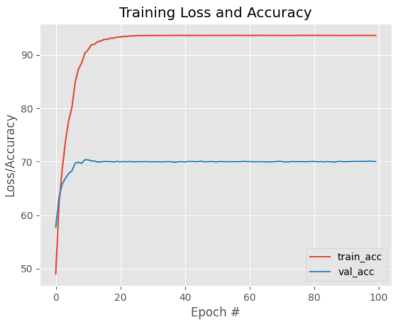
```
 [INFO] using device cuda
NMTModel(
  (encoder): NMTEncoder(
    (source_embedding): Embedding(3025, 64, padding_idx=0)
    (birnn): GRU(64, 32, batch_first=True, bidirectional=True)
  )
  (decoder): NMTDecoder(
    (target_embedding): Embedding(4911, 64, padding_idx=0)
    (gru_cell): GRUCell(128, 64)
    (hidden_map): Linear(in_features=64, out_features=64, bias=True)
    (classifier): Linear(in_features=128, out_features=4911, bias=True)
  )
)
```

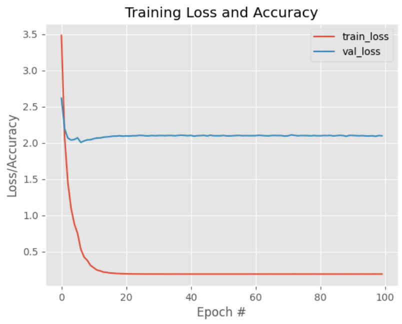
[INFO] epoch 19, **train loss 1.431**7084023827007, **val loss 2.64**23677929112173
[INFO] **train_acc 67.99**5278471898836, **val acc 59.98**05141023637784
[INFO] best validation loss updated and checkpoint saved

Evaluate NMT
[INFO] **mean** result: **0.359**5686680308878, **median** result**: 0.3328**359601570994

Comentários:
Melhor resultado até aqui, contudo temos muito overfitting. Vamos aumentar a taxa de aprendizagem.

4 **Hyperparâmetros:**
source_embedding_size=64
target_embedding_size=64
encoding_size=32
batch_size=32
num_epochs=100
learning_rate=**0.005**





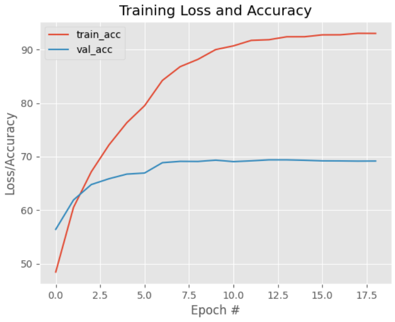
```
 [INFO] using device cuda
NMTModel(
  (encoder): NMTEncoder(
    (source_embedding): Embedding(3025, 64, padding_idx=0)
    (birnn): GRU(64, 32, batch_first=True, bidirectional=True)
  )
  (decoder): NMTDecoder(
    (target_embedding): Embedding(4911, 64, padding_idx=0)
    (gru_cell): GRUCell(128, 64)
    (hidden_map): Linear(in_features=64, out_features=64, bias=True)
    (classifier): Linear(in_features=128, out_features=4911, bias=True)
  )
)
```

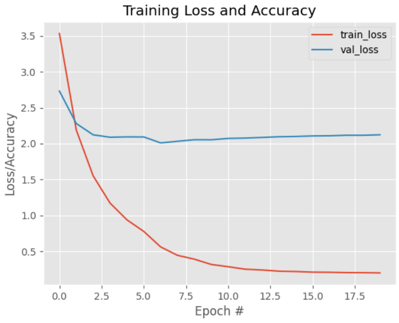
[INFO] epoch 6, **train loss 0.530**2001076832155, **val loss 2.00**58176751996646
[INFO] **train_acc 84.79**429008294372, **val acc 69.72**04647070264
[INFO] best validation loss updated and checkpoint saved

Evaluate NMT
[INFO] **mean** result: **0.545**4695553108071, **median** result: **0.5439**094815855025

Comentários:
Melhor resultado alcançado, contudo o overfitting não diminuiu. Habilitaremos a CrossEntropyLoss**.**

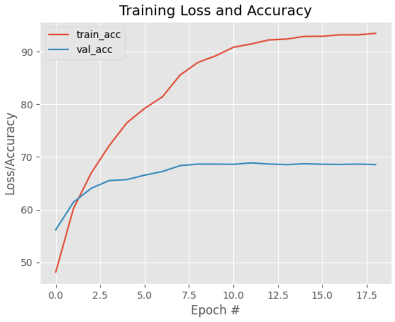
5 **Hyperparâmetros:**
source_embedding_size=64
target_embedding_size=64
encoding_size=32
batch_size=32
num_epochs=100
learning_rate=0.005
loss_func =
nn.CrossEntropyLoss()

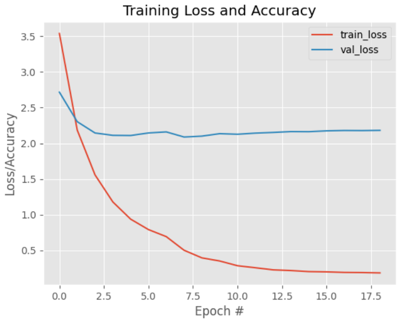




Comentários:
Paramos o treinamneto na época 18, pois não observamos nehuma melhoria no overfitting. Vamos habilitar o method 2: sample from distribution

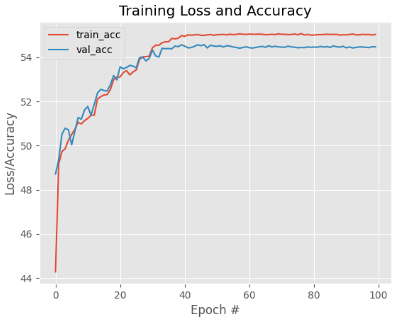
6 **Hyperparâmetros:**
source_embedding_size=64
target_embedding_size=64
encoding_size=32
batch_size=32
num_epochs=100
learning_rate=0.005
loss_func = nn.CrossEntropyLoss()
# method 2: sample from distribution
candidate_input = torch.multinomial(p_y_t_index, 1).squeeze()

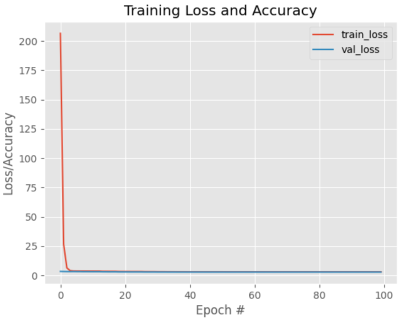




Comentários:
Paramos o treinamneto na época 18, pois não observamos nehuma melhoria no overfitting. Vamos habilitar a regularização L2.

7 **Hyperparâmetros:**
source_embedding_size=64
target_embedding_size=64
encoding_size=32
batch_size=32
num_epochs=100
learning_rate=0.005
l2_regularization = **0.001**





```
 [INFO] using device cpu
NMTModel(
  (encoder): NMTEncoder(
    (source_embedding): Embedding(3025, 64, padding_idx=0)
    (birnn): GRU(64, 32, batch_first=True, bidirectional=True)
  )
  (decoder): NMTDecoder(
    (target_embedding): Embedding(4911, 64, padding_idx=0)
    (gru_cell): GRUCell(128, 64)
    (hidden_map): Linear(in_features=64, out_features=64, bias=True)
    (classifier): Linear(in_features=128, out_features=4911, bias=True)
  )
)
[INFO] epoch 45, train loss 3.076054169002329, val loss 2.8562438409836557
[INFO] train_acc 54.99467830925427, val acc 54.526114394602814
[INFO] best validation loss updated and checkpoint saved
```

Evaluate NMT

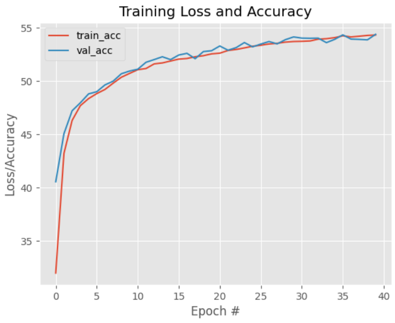
```
[INFO] mean result: 0.33115399770341614, median result: 0.3219038573439098
```

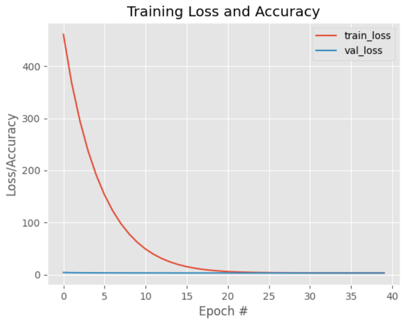
Comentários:

Diminuimos o overfitting, mas a acurácia diminui expressivamente.

8 **Hyperparâmetros:**

source_embedding_size=64
target_embedding_size=64
encoding_size=32
batch_size=32
num_epochs=40
learning_rate=**0.0005**
l2_regularization = 0.001





```
[INFO] using device cpu
NMTModel(
  (encoder): NMTEncoder(
    (source_embedding): Embedding(3025, 64, padding_idx=0)
    (birnn): GRU(64, 32, batch_first=True, bidirectional=True))
  (decoder): NMTDecoder(
    (target_embedding): Embedding(4911, 64, padding_idx=0)
    (gru_cell): GRUCell(128, 64)
    (hidden_map): Linear(in_features=64, out_features=64, bias=True)
    (classifier): Linear(in_features=128, out_features=4911, bias=True)))

[INFO] epoch 39, train loss 3.231030480066935, val loss 2.9168540337046633
[INFO] train_acc 54.33450864742833, val acc 54.39915909182199
[INFO] best validation loss updated and checkpoint saved
```

Evaluate NMT

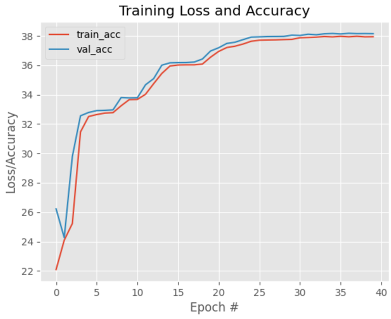
```
[INFO] mean result: 0.3217770244052043, median result: 0.3063079764817904
```

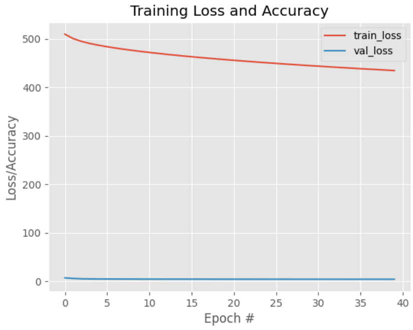
Comentários:

Diminuimos o overfitting, mas a acurácia não melhorou.

9 Hyperparâmetros:

```
source_embedding_size=64
target_embedding_size=64
encoding_size=32
batch_size=32
num_epochs=40
learning_rate=0.0005
l2_regularization = 0.001

loss_func = nn.CrossEntropyLoss()
optimizer =
optim.Adagrad(model.parameters(),
                    lr=learning_rate,
                    weight_decay=l2_regularization)
scheduler = optim.lr_scheduler.ReduceLROnPlateau
(optimizer=optimizer, mode='min', factor=0.5, patience=1)
```



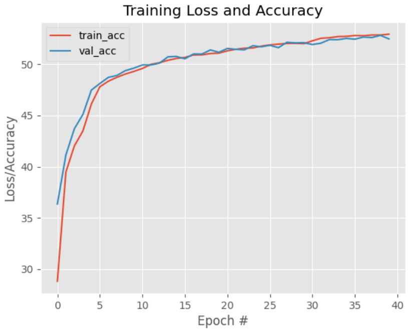


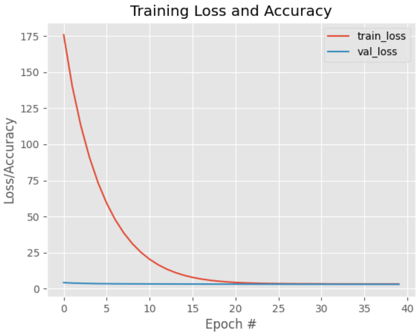
```
[INFO] using device cpu
NMTModel(
(encoder): NMTEncoder(
(source_embedding): Embedding(3025, 64, padding_idx=0)
(birnn): GRU(64, 32, batch_first=True, bidirectional=True)
)
(decoder): NMTDecoder(
(target_embedding): Embedding(4911, 64, padding_idx=0)
(gru_cell): GRUCell(128, 64)
(hidden_map): Linear(in_features=64, out_features=64, bias=True)
(classifier): Linear(in_features=128, out_features=4911, bias=True)))

[INFO] epoch 0, train loss 509.4013233920982, val loss 6.958291835472233
[INFO] train_acc 22.09021296112598, val acc 26.224576409561
[INFO] best validation loss updated and checkpoint saved
[INFO] epoch 39, train loss 434.5768570749383, val loss 4.123427863980903
[INFO] train_acc 37.94451676227942, val acc 38.15299286013642
[INFO] best validation loss updated and checkpoint saved

[INFO] mean result: 0.08892326948954755, median result: 0.0439975593063474
```

Comentário: Foi mudado o otimizador ADAM por Adagrad, mas a perda no treinamento não caiu, e a acurácia também não melhorou, vamos voltar para o ADAM e diminuir o tamanho do embedding_size

10 Hyperparâmetros:

```
source_embedding_size=24
target_embedding_size=24
encoding_size=32
batch_size=32
num_epochs=40
learning_rate=0.0005
l2_regularization = 0.001

loss_func = nn.CrossEntropyLoss()
optimizer =
optim.Adam(model.parameters(),
                    lr=learning_rate,
                    weight_decay=l2_regularization)
scheduler = optim.lr_scheduler.ReduceLROnPlateau
(optimizer=optimizer, mode='min', factor=0.5, patience=1)
```

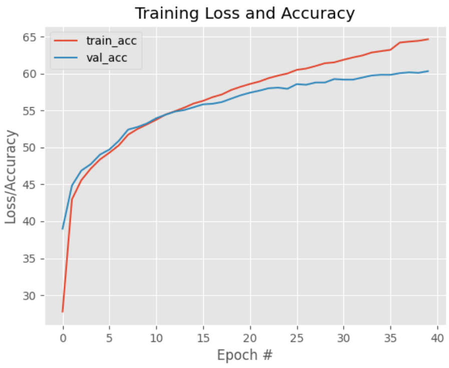


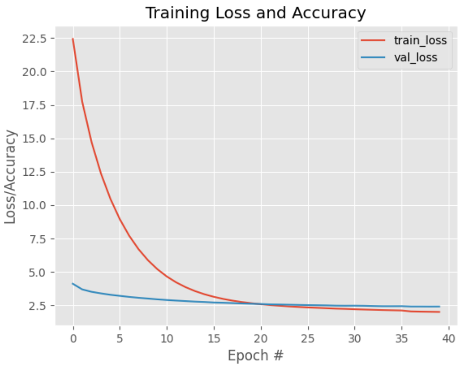


```
 [INFO] using device cpu
NMTModel(
  (encoder): NMTEncoder(
    (source_embedding): Embedding(3025, 24, padding_idx=0)
    (birnn): GRU(24, 32, batch_first=True, bidirectional=True)
  )
  (decoder): NMTDecoder(
    (target_embedding): Embedding(4911, 24, padding_idx=0)
    (gru_cell): GRUCell(88, 64)
    (hidden_map): Linear(in_features=64, out_features=64, bias=True)
    (classifier): Linear(in_features=128, out_features=4911, bias=True)
  )
)

[INFO] epoch 0, train loss 175.9322303102728, val loss 4.200261608498997
[INFO] train_acc 28.806423796941488, val acc 36.35309480582181
[INFO] epoch 39, train loss 3.265248627411694, val loss 3.0413855255627245
[INFO] train_acc 52.93673662988597, val acc 52.47779200615325

[INFO] mean result: 0.28951629576458243, median result: 0.28023947440512487
```

11 Hyperparâmetros:

```
source_embedding_size=24
target_embedding_size=24
encoding_size=32
batch_size=32
num_epochs=40
learning_rate=0.0005
l2_regularization = 0.0001

loss_func = nn.CrossEntropyLoss()
optimizer =
optim.Adam(model.parameters(),
                    lr=learning_rate,

weight_decay=l2_regularization)
scheduler = optim.lr_scheduler.ReduceLROnPlateau
(optimizer=optimizer, mode='min', factor=0.5, patience=1)
```





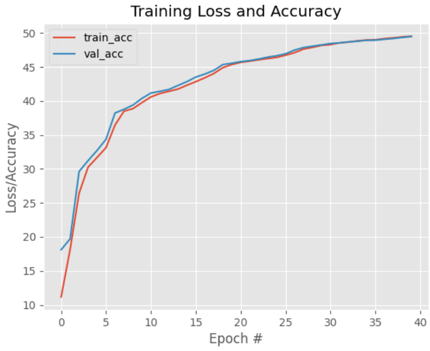
```
[INFO] using device cpu
NMTModel(
  (encoder): NMTEncoder(
    (source_embedding): Embedding(3025, 24, padding_idx=0)
    (birnn): GRU(24, 32, batch_first=True, bidirectional=True)
  )
  (decoder): NMTDecoder(
    (target_embedding): Embedding(4911, 24, padding_idx=0)
    (gru_cell): GRUCell(88, 64)
    (hidden_map): Linear(in_features=64, out_features=64, bias=True)
    (classifier): Linear(in_features=128, out_features=4911, bias=True)
  )
)

[INFO] epoch 0, train loss 22.4345690208569, val loss 4.12170874486204
[INFO] train_acc 27.7706076071854, val acc 38.98964887069802
[INFO] epoch 39, train loss 2.012769417595447, val loss 2.418139670715957
[INFO] train_acc 64.64790593242903, val acc 60.32771530425084


[INFO] mean result: 0.3684211164448593, median result: 0.3547357381807338
```

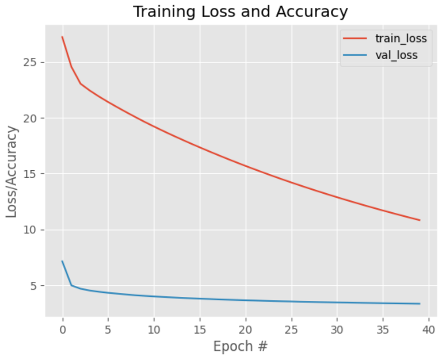
Comentário: O início da perda de treinamento começou com um bom parâmetro em comparação com os outros testes. Houve uma melhora no resultado final, tanto na perda validação e treinamento, e as acurácias também melhoraram, porém começou a ter overfiting. Vamos diminuir o learning rate

12 Hyperparâmetros:
source_embedding_size=24
target_embedding_size=24
encoding_size=32
batch_size=32
num_epochs=40
learning_rate=**0.00005**
l2_regularization = 0.0001

loss_func = nn.CrossEntropyLoss()
optimizer =
optim.**Adam**(model.parameters(),
lr=learning_rate,

weight_decay=l2_regularization)
scheduler = optim.lr_scheduler.ReduceLROnPlateau
(optimizer=optimizer, mode='min', factor=0.5, patience=1)

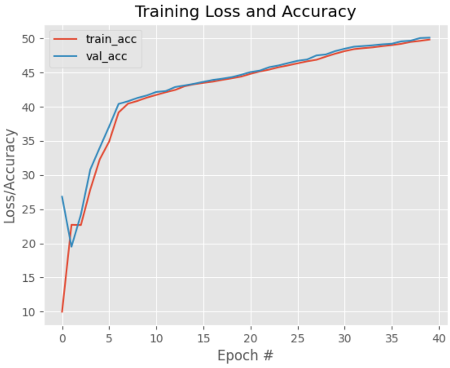


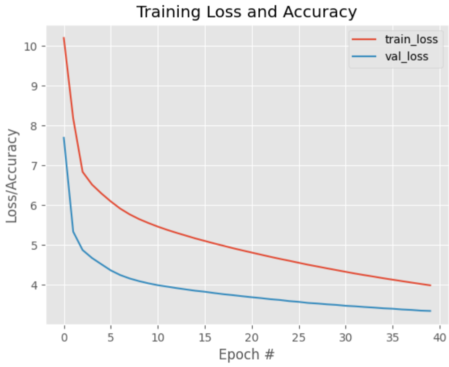


```
[INFO] using device cpu
NMTModel(
  (encoder): NMTEncoder(
    (source_embedding): Embedding(3025, 24, padding_idx=0)
    (birnn): GRU(24, 32, batch_first=True, bidirectional=True)
  )
  (decoder): NMTDecoder(
    (target_embedding): Embedding(4911, 24, padding_idx=0)
    (gru_cell): GRUCell(88, 64)
    (hidden_map): Linear(in_features=64, out_features=64, bias=True)
    (classifier): Linear(in_features=128, out_features=4911, bias=True)
  )
)

[INFO] epoch 0, train loss 27.216734180115814, val loss 7.1493658237769955
[INFO] train_acc 11.153796903258474, val acc 18.10693371622847
[INFO] train_acc 49.53038416246259, val acc 49.47142778088714
[INFO] best validation loss updated and checkpoint saved

[INFO] mean result: 0.2876296118027262, median result: 0.274651895686728
```

Comentário: Houve melhora no overfiting porém os resultados finais pioraram, por enquanto o melhor resultado foi o **teste 11**

13 Hyperparâmetros:
source_embedding_size=24
target_embedding_size=24
encoding_size=32
batch_size=32
num_epochs=40
learning_rate=0.00005
l2_regularization = **0.00001**





```
[INFO] using device cpu
NMTModel(
  (encoder): NMTEncoder(
    (source_embedding): Embedding(3025, 24, padding_idx=0)
    (birnn): GRU(24, 32, batch_first=True, bidirectional=True)
  )
  (decoder): NMTDecoder(
    (target_embedding): Embedding(4911, 24, padding_idx=0)
    (gru_cell): GRUCell(88, 64)
    (hidden_map): Linear(in_features=64, out_features=64, bias=True)
    (classifier): Linear(in_features=128, out_features=4911, bias=True)
  )
)
```

```
[INFO] train_acc 9.98763488656869, val acc 26.826147907940516
[INFO] best validation loss updated and checkpoint saved
[INFO] epoch 39, train loss 3.9877724538769663, val loss 3.343913543419759
[INFO] train_acc 49.83101039829001, val acc 50.10893205099904

[INFO] mean result: 0.2835466244526874, median result: 0.266633273235631
```

Comentário: Aumentando o learning rate melhorou os parâmetros de de perda, mas não teve impacto significativo na acurácia. Mais uma vez, o melhor treinamento foi o **teste 11**