Treinamento de um modelo T5 para traduzir de inglês para português usando dataset do Paracrawl.

Inicialização

Definindo os parâmetros, instalando e importando as bibliotecas.

```
# Configurações gerais
model_name = "t5-small"
batch_size = 32
accumulate_grad_batches = 2
source_max_length = 128
target_max_length = 128
learning_rate = 5e-3
```

```
! pip install sacrebleu
! pip install pytorch-lightning
! pip install transformers
```

```
oses (setup.py) ...
```

```
# Importar todos os pacotes de uma só vez para evitar duplicados ao
longo do notebook.
import gzip
import nvidia_smi
import os
import pytorch_lightning as pl
import random
import sacrebleu
import torch
import torch.nn.functional as F
import torch.nn as nn

from google.colab import drive

from pytorch_lightning.callbacks import ModelCheckpoint

from transformers import T5ForConditionalGeneration
from transformers import T5Tokenizer
from torch.utils.data import DataLoader
from torch.utils.data import Dataset

from typing import Dict
from typing import List
from typing import Tuple
```

```
seed = 123
random.seed(seed)
# np.random.seed(seed)
```

```
torch.random.manual_seed(seed)
torch.cuda.manual_seed(seed)

print(f"Pytorch Lightning Version: {pl.__version__}")
nvidia_smi.nvmlInit()
handle = nvidia_smi.nvmlDeviceGetHandleByIndex(0)
print(f"Device name: {nvidia_smi.nvmlDeviceGetName(handle)}")

def gpu_usage():
    global handle
    return str(nvidia_smi.nvmlDeviceGetUtilizationRates(handle).gpu) + 
'%'
```

```
Pytorch Lightning Version: 1.0.3
Device name: b'Tesla T4'
```

Iremos salvar os checkpoints (pesos do modelo) no google drive, para que possamos continuar o treino de onde paramos.

```
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly
remount, call drive.mount("/content/drive", force_remount=True).
```

Preparando Dados

Primeiro, fazemos download do dataset disponível nos arquivos da disciplina:

```
! wget -nc
https://storage.googleapis.com/neuralresearcher_data/unicamp/ia376e_20
20s1/paracrawl_enpt_train.tsv.gz
! wget -nc
https://storage.googleapis.com/neuralresearcher_data/unicamp/ia376e_20
20s1/paracrawl_enpt_test.tsv.gz
```

```
File ‘paracrawl_enpt_train.tsv.gz’ already there; not retrieving.

File ‘paracrawl_enpt_test.tsv.gz’ already there; not retrieving.
```

Dataset

Criaremos uma divisão de treino (100k pares) e val (5k pares) artificialmente.

```
def load_text_pairs(path):
    text_pairs = []
    for line in gzip.open(path, mode='rt'):
        text_pairs.append(line.strip().split('\t'))
    return text_pairs
```

```python
x_train = load_text_pairs('paracrawl_enpt_train.tsv.gz')
x_test = load_text_pairs('paracrawl_enpt_test.tsv.gz')

# Embaralhamos o treino para depois fazermos a divisão treino/val.
random.shuffle(x_train)

# Truncamos o dataset para 100k pares de treino e 5k pares de
validação.
x_val = x_train[100000:105000]
x_train = x_train[:100000]

for set_name, x in [('treino', x_train), ('validação', x_val),
('test', x_test)]:
    print(f'\n{len(x)} amostras de {set_name}')
    print(f'3 primeiras amostras {set_name}:')
    for i, (source, target) in enumerate(x[:3]):
        print(f'{i}: source: {source}\n   target: {target}')
```

```
100000 amostras de treino
3 primeiras amostras treino:
0: source: More Croatian words and phrases
   target: Mais palavras e frases em croata
1: source: Jerseys and pullovers, containing at least 50Â % by weight
of wool and weighing 600Â g or more per article 6110 11 10 (PCE)
   target: Camisolas e pulôveres, com pelo menos 50 %, em peso, de lã
e pesando 600g ou mais por unidade 6110 11 10 (PCE)
2: source: Atex Colombia SAS makes available its lead product, 100%
natural liquid latex, excellent quality and price. ... Welding
manizales caldas Colombia a DuckDuckGo
   target: Atex Colômbia SAS torna principal produto está disponível,
látex líquido 100% natural, excelente qualidade e preço. ...

5000 amostras de validação
3 primeiras amostras validação:
0: source: «You have hidden these things from the wise and the learned
you have revealed them to the childlike»
   target: «Escondeste estas coisas aos sábios e entendidos e as
revelaste aos pequenos»
1: source: Repair of computers, application programming, network
installations, web design, graphic design, and also the most. Computer
consulting in Santa Lucía
   target: Reparação de computadores, programação de aplicações,
instalações de rede, web design, design gráfico, e também a.
2: source: He was born in Fafe (Braga) and he graduated in Law in
Coimbra University.
   target: É natural de Fafe (Braga) e Licenciado em Direito pela
Universidade de Coimbra.

20000 amostras de test
```

```
3 primeiras amostras test:
0: source: In this way, the civil life of a nation matures, making it
possible for all citizens to enjoy the fruits of genuine tolerance and
mutual respect.
   target: Deste modo, a vida civil de uma nação amadurece, fazendo
com que todos os cidadãos gozem dos frutos da tolerância genuína e do
respeito mútuo.
1: source: 1999 XIII. Winnipeg, Canada July 23 to August 8
   target: 1999 XIII. Winnipeg, Canadá 23 de julho a 8 de agosto
2: source: In the mystery of Christmas, Christ's light shines on the
earth, spreading, as it were, in concentric circles.
   target: No mistério do Natal, a luz de Cristo irradia-se sobre a
terra, difundindo-se como círculos concêntricos.
```

Criando o DataLoader

```
tokenizer = T5Tokenizer.from_pretrained(model_name)
extra_tokens = 'Ã,Õ,Á,É,Í,Ó,Ú,À,ã,õ,á,é,í,ó,ú'.split(',')
added_tokens = []
for token in extra_tokens:
    enc = tokenizer.encode(token)
    if 2 in enc:
        added_tokens.append(token)
        tokenizer.add_tokens(token)

class MyDataset(Dataset):
    def __init__(self, text_pairs: List[Tuple[str]], tokenizer,
                 source_max_length: int = 32, target_max_length: int =
32):
        self.tokenizer = tokenizer
        self.text_pairs = text_pairs
        self.source_max_length = source_max_length
        self.target_max_length = target_max_length

        self.task_string = 'translate English to Portuguese: '

    def __len__(self):
        return len(self.text_pairs)

    def __getitem__(self, idx):
        source, target = self.text_pairs[idx]

        source = self.task_string + source

        tokens = tokenizer(source, padding='max_length',
truncation=True, max_length=self.source_max_length,
return_tensors="pt")
        source_token_ids = torch.squeeze(tokens['input_ids'], dim=0)
        source_mask = torch.squeeze(tokens['attention_mask'], dim=0)

        tokens = tokenizer(target, padding='max_length',
truncation=True, max_length=self.target_max_length,
return_tensors="pt")
        target_token_ids = torch.squeeze(tokens['input_ids'], dim=0)
        target_mask = torch.squeeze(tokens['attention_mask'], dim=0)

        return (source_token_ids, source_mask, target_token_ids,
target_mask,
                source, target)
```

Testando o DataLoader

```
text_pairs = [('we like pizza', 'eu gosto de pizza')]
dataset_debug = MyDataset(
    text_pairs=text_pairs,
    tokenizer=tokenizer,
    source_max_length=source_max_length,
    target_max_length=target_max_length)

dataloader_debug = DataLoader(dataset_debug, batch_size=10,
shuffle=True,
                              num_workers=0)

source_token_ids, source_mask, target_token_ids, target_mask, source,
target = next(iter(dataloader_debug))
print('source_token_ids:\n', source_token_ids)
print('source_mask:\n', source_mask)
print('target_token_ids:\n', target_token_ids)
print('target_mask:\n', target_mask)
print('source:\n', source)
print('target:\n', target)

print('source_token_ids.shape:', source_token_ids.shape)
print('source_mask.shape:', source_mask.shape)
print('target_token_ids.shape:', target_token_ids.shape)
print('target_mask.shape:', target_mask.shape)

source_token_ids:
 tensor([[13959,  1566,    12, 21076,    10,    62,   114,  6871,
1,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0,
0,     0,
             0,     0,     0,     0,     0,     0,     0,     0]])
source_mask:
 tensor([[1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0]])
target_token_ids:
 tensor([[   3,   15,   76,  281,    7,  235,   20, 6871,    1,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
0,    0,
            0,    0,    0,    0,    0,    0,    0,    0]])
target_mask:
 tensor([[1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0]])
source:
 ('translate English to Portuguese: we like pizza',)
target:
 ('eu gosto de pizza',)
source_token_ids.shape: torch.Size([1, 128])
source_mask.shape: torch.Size([1, 128])
target_token_ids.shape: torch.Size([1, 128])
target_mask.shape: torch.Size([1, 128])
```

Instanciando os DataLoaders de Treino/Val/Test

```
dataset_train = MyDataset(text_pairs=x_train,
                          tokenizer=tokenizer,
                          source_max_length=source_max_length,
                          target_max_length=target_max_length)

dataset_val = MyDataset(text_pairs=x_val,
                        tokenizer=tokenizer,
                        source_max_length=source_max_length,
                        target_max_length=target_max_length)

dataset_test = MyDataset(text_pairs=x_test,
                         tokenizer=tokenizer,
                         source_max_length=source_max_length,
                         target_max_length=target_max_length)

train_dataloader = DataLoader(dataset_train, batch_size=batch_size,
                              shuffle=True, num_workers=4)

val_dataloader = DataLoader(dataset_val, batch_size=batch_size,
shuffle=False,
                            num_workers=4)

test_dataloader = DataLoader(dataset_test, batch_size=batch_size,
                             shuffle=False, num_workers=4)
```

T5 com Pytorch Lightning

```
class T5Finetuner(pl.LightningModule):

    def __init__(self, tokenizer, train_dataloader, val_dataloader,
                 test_dataloader, learning_rate,
target_max_length=32):
        super(T5Finetuner, self).__init__()

        self._train_dataloader = train_dataloader
        self._val_dataloader = val_dataloader
        self._test_dataloader = test_dataloader

        self.model = 
T5ForConditionalGeneration.from_pretrained(model_name, 
return_dict=True)

        self.tokenizer = tokenizer
        self.learning_rate = learning_rate
        self.target_max_length = target_max_length

        self.print = 0

    def forward(self, source_token_ids, source_mask, 
target_token_ids=None,
                target_mask=None):

        if self.training:
            encoder_hidden_states = self.model.get_encoder()
(source_token_ids,

attention_mask=source_mask)

            target_token_ids[target_mask == 0] = -100
            output = self.model(encoder_outputs=encoder_hidden_states,
                                attention_mask=source_mask,
                                labels=target_token_ids)
            return output.loss
        else:
            with torch.no_grad():
                predicted_token_ids = 
self.model.generate(input_ids=source_token_ids,

max_length=self.target_max_length,

do_sample=False).squeeze()
                return predicted_token_ids

    def training_step(self, batch, batch_nb):
        # batch
        source_token_ids, source_mask, target_token_ids, target_mask, 
_, _ = batch

        # fwd
        loss = self(
            source_token_ids, source_mask, target_token_ids,
target_mask)

        # logs
        tensorboard_logs = {'train_loss': loss}
        progress_bar = {'gpu_usage': gpu_usage()}
        return {'loss': loss, 'log': tensorboard_logs, 'progress_bar':
progress_bar}

    def validation_step(self, batch, batch_nb):
        self.training = False
        source_token_ids, source_mask, target_token_ids, target_mask,
source, target = batch

        predicted_ids = self(source_token_ids, source_mask,
target_token_ids, target_mask)
        B = []
        i = 0
        for pred_ids, targ in zip(predicted_ids, target):
            predicted = self.tokenizer.decode(pred_ids,
skip_special_tokens=True)
            bleu = sacrebleu.corpus_bleu(predicted, targ)
            B.append(bleu.score)

            if i < 1 and self.print % 100 == 0:
                print(f'\nsource = {source[i]}')
                print(f'target = {target[i]}')
                print(f'predicted = {predicted}')
            i += 1

        self.print += 1

        B_pt = torch.tensor(B)
        avg_bleu = torch.mean(B_pt)

        progress_bar = {'gpu_usage': gpu_usage()}
        return {'val_bleu': avg_bleu, 'progress_bar': progress_bar}

    def test_step(self, batch, batch_nb):
        self.training = False
        source_token_ids, source_mask, target_token_ids, target_mask,
source, target = batch

        predicted_ids = self(source_token_ids, source_mask,
target_token_ids, target_mask)
        B = []
        i = 0
        for pred_ids, targ in zip(predicted_ids, target):
            predicted = self.tokenizer.decode(pred_ids,
skip_special_tokens=True)
```

```
            bleu = sacrebleu.corpus_bleu(predicted, targ)
            B.append(bleu.score)

            if i < 1 and self.print % 100 == 0:
                print(f'\nsource = {source[i]}')
                print(f'target = {target[i]}')
                print(f'predicted = {predicted}')
            i += 1

        self.print += 1

        B_pt = torch.tensor(B)
        avg_bleu = torch.mean(B_pt)

        progress_bar = {'gpu_usage': gpu_usage()}
        return {'test_bleu': avg_bleu, 'progress_bar': progress_bar}

    def validation_epoch_end(self, outputs):
        avg_bleu = sum([x['val_bleu'] for x in outputs]) /
len(outputs)

        tensorboard_logs = {'avg_val_bleu': avg_bleu}

        return {'avg_val_bleu': avg_bleu, 'progress_bar':
tensorboard_logs}

    def test_epoch_end(self, outputs):
        avg_bleu = sum([x['test_bleu'] for x in outputs]) /
len(outputs)

        tensorboard_logs = {'avg_test_bleu': avg_bleu}

        return {'avg_test_bleu': avg_bleu, 'progress_bar':
tensorboard_logs}

    def configure_optimizers(self):
        return torch.optim.Adam(
            [p for p in self.parameters() if p.requires_grad],
            lr=self.learning_rate, eps=1e-08)

    def train_dataloader(self):
        return self._train_dataloader

    def val_dataloader(self):
        return self._val_dataloader

    def test_dataloader(self):
        return self._test_dataloader
```

Instanciando o T5

```
model = T5Finetuner(tokenizer=tokenizer,
                    train_dataloader=train_dataloader,
                    val_dataloader=val_dataloader,
                    test_dataloader=test_dataloader,
                    learning_rate=learning_rate,
                    target_max_length=target_max_length)
```

Número de parâmetros do modelo

```
sum([torch.tensor(x.size()).prod() for x in model.parameters() if
x.requires_grad]) # trainable parameters
```

```
tensor(60506880)
```

Debug

Testando rapidamente o modelo em treino, validação e teste com um batch

```
trainer = pl.Trainer(gpus=1,
                     checkpoint_callback=False,  # Disable checkpoint
saving.
                     fast_dev_run=True)
trainer.fit(model)
trainer.test(model)
del model  # Para não ter estouro de mémoria da GPU
```

```
GPU available: True, used: True
TPU available: False, using: 0 TPU cores
LOCAL_RANK: 0 - CUDA_VISIBLE_DEVICES: [0]
Running in fast_dev_run mode: will run a full train, val and test loop
using a single batch

  | Name  | Type                       | Params
-----------------------------------------------------
0 | model | T5ForConditionalGeneration | 60 M
```

{"version_major":2,"version_minor":0,"model_id":"c8731b2851ef449dbb5e3
cda9d859ae2"}

```
/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The {log:dict keyword} was deprecated
in 0.9.1 and will be removed in 1.0.0
Please use self.log(...) inside the lightningModule instead.

# log on a step or aggregate epoch metric to the logger and/or
progress bar
# (inside LightningModule)
self.log('train_loss', loss, on_step=True, on_epoch=True,
prog_bar=True)
```

```
  warnings.warn(*args, **kwargs)
/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The {progress_bar:dict keyword} was
deprecated in 0.9.1 and will be removed in 1.0.0
Please use self.log(...) inside the lightningModule instead.

# log on a step or aggregate epoch metric to the logger and/or
progress bar
# (inside LightningModule)
self.log('train_loss', loss, on_step=True, on_epoch=True,
prog_bar=True)
  warnings.warn(*args, **kwargs)
```

{"version_major":2,"version_minor":0,"model_id":"b65148050dd648b9b522b
4b337474c38"}

```
source = translate English to Portuguese: «You have hidden these
things from the wise and the learned you have revealed them to the
childlike»
target = «Escondeste estas coisas aos sábios e entendidos e as
revelaste aos pequenos»
predicted = «Vous a caché ces choses de la sa e apprendt vous ve
révéle e a e a e a e a e a e a e a e a e a e a e a e e e e e e e e e e e
e e e e e e e e e e e e e aveve ca ca as e s

/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The validation_epoch_end should not
return anything as of 9.1.to log, use self.log(...) or self.write(...)
directly in the LightningModule
  warnings.warn(*args, **kwargs)
```

{"version_major":2,"version_minor":0,"model_id":"1519ad186a03410db3077
6bb77ea379e"}

```
----------------------------------------------------------------------
----------
DATALOADER:0 TEST RESULTS
{'avg_test_bleu': tensor(7.2845)}
----------------------------------------------------------------------
----------


/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The testing_epoch_end should not
return anything as of 9.1.to log, use self.log(...) or self.write(...)
directly in the LightningModule
  warnings.warn(*args, **kwargs)
```

Overfit em algumas amostras

Antes de treinar o modelo no dataset todo, faremos overfit do modelo em poucas de treino para verificar se loss vai para próximo de 0. Isso serve para depurar se a implementação do modelo está correta.

Podemos também medir se a acurácia neste minibatch chega perto de 100%. Isso serve para depurar se nossa função que mede a acurácia está correta.

Nota: se treinarmos por muitas épocas (ex: 500) é possivel que a loss vá para zero mesmo com bugs na implementação. O ideal é que a loss chege próxima a zero antes de 100 épocas.

```
trainer = pl.Trainer(gpus=1,
                     max_epochs=30,
                     check_val_every_n_epoch=10,
                     checkpoint_callback=False, # Disable checkpoint 
saving
                     overfit_batches=0.005)


# Dataset usando apenas um batch de amostras de treino.
dataset_debug = MyDataset(text_pairs=x_train,
                          tokenizer=tokenizer,
                          source_max_length=source_max_length,
                          target_max_length=target_max_length)

debug_dataloader = DataLoader(dataset_debug, batch_size=batch_size,
                              shuffle=False, num_workers=4)

model = T5Finetuner(tokenizer=tokenizer,
                    train_dataloader=debug_dataloader,
                    val_dataloader=debug_dataloader,
                    test_dataloader=None,
                    learning_rate=learning_rate,
                    target_max_length=target_max_length)

trainer.fit(model)
del model  # Para não ter estouro de mémoria da GPU
```

```
GPU available: True, used: True
TPU available: False, using: 0 TPU cores
LOCAL_RANK: 0 - CUDA_VISIBLE_DEVICES: [0]

  | Name  | Type                       | Params
-----------------------------------------------------
0 | model | T5ForConditionalGeneration | 60 M
```

{"version_major":2,"version_minor":0,"model_id":"a0f484ab0dff4b06b9ae0
f87de43f350"}

```
source = translate English to Portuguese: More Croatian words and
phrases
target = Mais palavras e frases em croata
predicted = Portugiesisch und kroatisch mehr Worte
```

```
/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The validation_epoch_end should not
return anything as of 9.1.to log, use self.log(...) or self.write(...)
directly in the LightningModule
  warnings.warn(*args, **kwargs)
/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The {progress_bar:dict keyword} was
deprecated in 0.9.1 and will be removed in 1.0.0
Please use self.log(...) inside the lightningModule instead.

# log on a step or aggregate epoch metric to the logger and/or
progress bar
# (inside LightningModule)
self.log('train_loss', loss, on_step=True, on_epoch=True,
prog_bar=True)
  warnings.warn(*args, **kwargs)
```

{"version_major":2,"version_minor":0,"model_id":"0bb289ddb1904f0984fc6
a5a56ddca47"}

```
/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The {log:dict keyword} was deprecated
in 0.9.1 and will be removed in 1.0.0
Please use self.log(...) inside the lightningModule instead.

# log on a step or aggregate epoch metric to the logger and/or
progress bar
# (inside LightningModule)
self.log('train_loss', loss, on_step=True, on_epoch=True,
prog_bar=True)
  warnings.warn(*args, **kwargs)
```

{"version_major":2,"version_minor":0,"model_id":"714887a58b2544cfb11a7
f997e657593"}

{"version_major":2,"version_minor":0,"model_id":"b667c45fba3a45e2a98df
41eb4041b3a"}

{"version_major":2,"version_minor":0,"model_id":"998b4d78067d45cc8e1e5
9150171e66e"}

Treinamento e Validação

```
max_epochs = 2

checkpoint_path = '/content/drive/My Drive/Colab Notebooks/Tópicos 
IA/Aula 5/checkpoints/epoch=10.ckpt'
checkpoint_dir = os.path.dirname(os.path.abspath(checkpoint_path))
print(f'Files in {checkpoint_dir}: {os.listdir(checkpoint_dir)}')
print(f'Saving checkpoints to {checkpoint_dir}')
checkpoint_callback = ModelCheckpoint(filepath=checkpoint_dir,
                                      save_top_k=-1)  # Keeps all 
checkpoints.

resume_from_checkpoint = None
if os.path.exists(checkpoint_path):
    print(f'Restoring checkpoint: {checkpoint_path}')
    resume_from_checkpoint = checkpoint_path

trainer = pl.Trainer(gpus=1,
                     max_epochs=max_epochs,
                     check_val_every_n_epoch=1,
                     profiler=True,
                     accumulate_grad_batches=accumulate_grad_batches,
                     checkpoint_callback=checkpoint_callback,
                     progress_bar_refresh_rate=50,
                     resume_from_checkpoint=resume_from_checkpoint)

model = T5Finetuner(tokenizer=tokenizer,
                    train_dataloader=train_dataloader,
                    val_dataloader=val_dataloader,
                    test_dataloader=test_dataloader,
                    learning_rate=learning_rate,
                    target_max_length=target_max_length)

trainer.fit(model)
```

```
GPU available: True, used: True
TPU available: False, using: 0 TPU cores
LOCAL_RANK: 0 - CUDA_VISIBLE_DEVICES: [0]

Files in /content/drive/My Drive/Colab Notebooks/Tópicos IA/Aula 
5/checkpoints: ['epoch=4.ckpt']
Saving checkpoints to /content/drive/My Drive/Colab Notebooks/Tópicos 
IA/Aula 5/checkpoints


  | Name  | Type                       | Params
-----------------------------------------------------
0 | model | T5ForConditionalGeneration | 60 M
```

{"version_major":2,"version_minor":0,"model_id":"1d57f0aa3b4b4f639f71fd4d36494aef"}

```
source = translate English to Portuguese: «You have hidden these
things from the wise and the learned you have revealed them to the
childlike»
target = «Escondeste estas coisas aos sábios e entendidos e as
revelaste aos pequenos»
predicted = «Du hast diese Dinge den Weisen und den gelernten Sie
haben sie dem Kindesenkung offenbart»
```

```
/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The validation_epoch_end should not
return anything as of 9.1.to log, use self.log(...) or self.write(...)
directly in the LightningModule
  warnings.warn(*args, **kwargs)
/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The {progress_bar:dict keyword} was
deprecated in 0.9.1 and will be removed in 1.0.0
Please use self.log(...) inside the lightningModule instead.

# log on a step or aggregate epoch metric to the logger and/or
progress bar
# (inside LightningModule)
self.log('train_loss', loss, on_step=True, on_epoch=True,
prog_bar=True)
  warnings.warn(*args, **kwargs)
```

{"version_major":2,"version_minor":0,"model_id":"4f9a1842345f4fdabfce9225cd547e01"}

```
/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The {log:dict keyword} was deprecated
in 0.9.1 and will be removed in 1.0.0
Please use self.log(...) inside the lightningModule instead.

# log on a step or aggregate epoch metric to the logger and/or
progress bar
# (inside LightningModule)
self.log('train_loss', loss, on_step=True, on_epoch=True,
prog_bar=True)
  warnings.warn(*args, **kwargs)
```

{"version_major":2,"version_minor":0,"model_id":"ed53e431000e4f8dbbae9207160b1685"}

```
source = translate English to Portuguese: Greeting of Card. Alfonso
López Trujillo during the vigil of prayer held in the City of Arts and
Sciences of Valencia (July 8, 2006)
```

```
target = Saudação do Cardeal Alfonso López Trujillo no início do
encontro de festa e de testemunho na Cidade das Artes e Ciências de
Valência (8 de julho de 2006)
predicted = Acolhimento de Cart ã o. Alfonso López Trujillo durante a
vig í lia da oraç ã o realizada na cidade das Artes e das Ciências de
Valencia (08 de julho de 2006)
```

{"version_major":2,"version_minor":0,"model_id":"a45aa06172684d908c9dd
d3da617aaf9"}

```
source = translate English to Portuguese: I am Brazilian and I work in
the Catholic school of Sainte-Marie.
target = Sou brasileira e trabalho na escola católica de Sainte-Marie.
predicted = Eu sou brasileiro e trabalho na escola católica de Sainte-
Marie.

source = translate English to Portuguese: Information: Located in San
Diego.
target = Informação: Located in San Diego.
predicted = Informaç õ es: Localizado em San Diego.



Profiler Report

Action                |  Mean duration (s)  |  Total time (s)
-----------------------------------------------------------------
on_fit_start          |  3.0092e-05         |  3.0092e-05
on_validation_start   |  0.019239           |  0.057717
on_validation_epoch_start  |  3.0087e-05         |  9.0262e-05
on_validation_batch_start  |  2.1706e-05         |  0.0068591
validation_step_end   |  2.322e-05          |  0.0073374
on_validation_batch_end    |  8.4344e-05         |  0.026653
on_validation_epoch_end    |  2.3875e-05         |  7.1625e-05
on_validation_end     |  2.976              |  8.9279
on_train_start        |  0.027495           |  0.027495
on_epoch_start        |  0.0023539          |  0.0047078
on_train_epoch_start  |  1.5032e-05         |  3.0065e-05
get_train_batch       |  0.0025898          |  16.186
on_batch_start        |  2.6563e-05         |  0.16602
on_train_batch_start  |  1.368e-05          |  0.085498
training_step_end     |  1.5593e-05         |  0.097458
model_forward         |  0.075873           |  474.21
model_backward        |  0.44769            |  2798.1
on_after_backward     |  2.6889e-05         |  0.16806
on_batch_end          |  2.4118e-05         |  0.15074
on_train_batch_end    |  9.8018e-05         |  0.61261
optimizer_step        |  0.012067           |  37.722
on_epoch_end          |  2.3869e-05         |  4.7739e-05
```

```
on_train_epoch_end    |  1.5048e-05         |  3.0097e-05
on_train_end          |  0.0030901          |  0.0030901
```

```
1
```

Teste

Após treinado, avaliamos o modelo no dataset de teste. É importante que essa avaliação seja feita poucas vezes para evitar "overfit manual" no dataset de teste.

```
trainer.test(model)
```

{"version_major":2,"version_minor":0,"model_id":"32b4edba9b6b4e06855b8f11ef74b555"}

```
source = translate English to Portuguese: There are lots of taxis in
Bangkok. It is relatively cheap and the cars often contain both air
conditioning and a meter.
target = Há lotes de táxis em Bangkok. É relativamente barato e os
carros muitas vezes contêm tanto ar condicionado e um medidor.
predicted = Há muitos táxis em Bangkok. É relativamente barata e os
carros frequentemente contêm ar condicionado e um metro.

source = translate English to Portuguese: Thanks to its formulation,
it is particularly suitable in the warm periods of the year or
when ...
target = Graças à sua formulação, é particularmente adequado nos
períodos quentes do ano ...
predicted = Graças à sua formulaç ã o, é particularmente adequada nos
per í odos quentes do ano ou quando...

source = translate English to Portuguese: Digitization of documents in
Manto (Olancho, Honduras) - Amarillashonduras.net
target = Digitalização de documentos em Manto (Olancho, Honduras) -
Amarillashonduras.net
predicted = Digitalizaç ã o de documentos em San Pedro de Macor í s
(Cortés, Honduras) - Amarillashonduras.net

source = translate English to Portuguese: Plastic injection in Santo
Domingo - AmarillasLatinas.net
target = Injeção de plástico em Santiago - AmarillasLatinas.net
predicted = Injeç ã o de plástico em Santo Domingo -
AmarillasLatinas.net

source = translate English to Portuguese: The protein is referred to
```

```
as the target of RAPAMYCIN due to the discovery that SIROLIMUS
(commonly known as rapamycin) forms an inhibitory complex with
TACROLIMUS BINDING PROTEIN 1A that blocks the action of its enzymatic
activity. History Note English: 2011
target = A proteína é conhecida por ser alvo da rapamicina devido à
descoberta de que o SIROLIMO (também conhecido como rapamicina) forma
um complexo inibitório com a PROTEÍNA 1A DE LIGAÇÃO A TACROLIMO que
bloqueia a ação de sua atividade enzimática.
predicted = A prote í na é referida como objetivo de RAPAMYCIN devido
ao descoberto que SIROLIMUS (conhecido como rapamycina) forma um
complexo inibidor com TACROLIMUS ABINDING PROTEIN 1A que bloquea a aç
ã o da sua atividade enc í matica.

source = translate English to Portuguese: Next: Mental health and
feminism are the themes of the 7th winners DOC Future Related Works
target = Próximo: Saúde mental e feminismo são os temas vencedores do
7º DOC Futura
predicted = Próximo: Sa ú de mental e feminismo s ã o os temas dos 7os
vencedores DOC futuro
----------------------------------------------------------------------
----------
DATALOADER:0 TEST RESULTS
{'avg_test_bleu': tensor(28.0566)}
----------------------------------------------------------------------
----------


/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The testing_epoch_end should not
return anything as of 9.1.to log, use self.log(...) or self.write(...)
directly in the LightningModule
  warnings.warn(*args, **kwargs)
/usr/local/lib/python3.6/dist-packages/pytorch_lightning/utilities/
distributed.py:45: UserWarning: The {progress_bar:dict keyword} was
deprecated in 0.9.1 and will be removed in 1.0.0
Please use self.log(...) inside the lightningModule instead.

# log on a step or aggregate epoch metric to the logger and/or
progress bar
# (inside LightningModule)
self.log('train_loss', loss, on_step=True, on_epoch=True,
prog_bar=True)
  warnings.warn(*args, **kwargs)

[{'avg_test_bleu': 28.05661010742187 5}]
```

```
!nvidia-smi
```

```
Fri Oct 23 00:06:44 2020
+---------------------------------------------------------------------
--------+
| NVIDIA-SMI 455.23.05    Driver Version: 418.67       CUDA Version:
10.1     |
|-------------------------------+----------------------
+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile
Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util
Compute M. |
|                               |                      |
MIG M. |
|
===============================+======================+===============
=======|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |
0 |
| N/A   76C    P0    46W /  70W |   3949MiB / 15079MiB |      0%
Default |
|                               |                      |
ERR! |
+-------------------------------+----------------------
+----------------------+


+----------------------------------------------------------------------
--------+
| Processes:
|
|  GPU   GI   CI        PID   Type   Process name                  GPU
Memory |
|        ID   ID
Usage      |
|
======================================================================
=======|
|  No running processes found
|
+----------------------------------------------------------------------
--------+
```