bart_title_2_storyline

December 1, 2021

1 Use Bart to generate storyline from storytitle

References to https://towardsdatascience.com/teaching-bart-to-rap-fine-tuning-hugging-faces-bart-model-41749d38f3ef

[ ]:
```
from google.colab import drive
drive.mount('/content/drive')
root_dir = "/content/drive/MyDrive/Colab Notebooks/544_bart/"
```

```
Mounted at /content/drive
```

[ ]:
```
# This run uses Pytorch Lightening to finetune the model
!pip install -q pytorch-lightning
!pip install -q transformers
```

[ ]:
```
# imports
import transformers
from torch.utils.data import DataLoader, TensorDataset, random_split,␣
 ↪RandomSampler, Dataset
import pandas as pd
import numpy as np

import torch.nn.functional as F
import pytorch_lightning as pl
import torch
from pytorch_lightning.callbacks import ModelCheckpoint

import math
import random
import re
import argparse
```

[ ]:
```
df = pd.read_csv(root_dir + 'train_title_line_story.csv')
df
```

[ ]:
```
                    storytitle  ...
titleLineConcat
0       David Drops the Weight  ...  David Drops the Weight <EOT> put try
realized ...
```

```
1                  Frustration  ...        Frustration <EOT> tom angry wall tom
regret
2           Marcus Buys Khakis  ...  Marcus Buys Khakis <EOT> business formal
pair ...
3           Different Opinions  ...  Different Opinions <EOT> trailer needed
much w...
4      Overcoming shortcomings  ...  Overcoming shortcomings <EOT> pastor tried
sin...
...                        ...  ...
...
52660                  Flavor  ...        Flavor <EOT> flavor tried get recipe
recipe
52661              After Death  ...      After Death <EOT> trouble day found
told week
52662  Janice breaks her wrist  ...  Janice breaks her wrist <EOT> janice legs
work...
52663   Jamie marries for love  ...  Jamie marries for love <EOT> jamie married
man...
52664                  Orange  ...               Orange <EOT> tree hit tree
tree nose

[52665 rows x 4 columns]
```

[ ]:
```
train, valid, test = np.split(df, [int(.8*len(df)), int(.9*len(df))])
```

[ ]:
```
train
```

[ ]:
```
                    storytitle  ...
titleLineConcat
0       David Drops the Weight  ...  David Drops the Weight <EOT> put try
realized ...
1                  Frustration  ...        Frustration <EOT> tom angry wall tom
regret
2           Marcus Buys Khakis  ...  Marcus Buys Khakis <EOT> business formal
pair ...
3           Different Opinions  ...  Different Opinions <EOT> trailer needed
much w...
4      Overcoming shortcomings  ...  Overcoming shortcomings <EOT> pastor tried
sin...
...                        ...  ...
...
42127        Long Disney lines  ...  Long Disney lines <EOT> went waited longest
le...
42128                    Ropes  ...             Ropes <EOT> took went shore rope
great
42129                Detention  ...  Detention <EOT> punished waiting write
upset time
42130       Forgotten Homework  ...  Forgotten Homework <EOT> ryan mom school
```

```
schoo...
42131              The Dresser  ...    The Dresser <EOT> shopping found took
room well

[42132 rows x 4 columns]
```

[ ]:
```
valid
```

[ ]:
```
                storytitle  ...
titleLineConcat
42132               Helper  ...         Helper <EOT> work flat pulled late
trouble
42133      Psychology Exam  ...  Psychology Exam <EOT> steve well studying
time...
42134         Old Memories  ...   Old Memories <EOT> year rob get searching
kissed
42135  Craigslist Shopping  ...  Craigslist Shopping <EOT> furniture money
look...
42136      Geese Migration  ...  Geese Migration <EOT> school road stop road
sc...
...                    ...  ...
...
47393        Free Sandwich  ...  Free Sandwich <EOT> chris use walked ordered
meal
47394                Barry  ...     Barry <EOT> lonely stays yesterday got
quickly
47395               Storms  ...     Storms <EOT> low turning time ground
terrified
47396               Mirror  ...             Mirror <EOT> mirror saw went hand
knew
47397             Nintendo  ...          Nintendo <EOT> yoshio play old yoshio
toy

[5266 rows x 4 columns]
```

[ ]:
```
test
```

[ ]:
```
                    storytitle  ...
titleLineConcat
47398                   Crazed  ...       Crazed <EOT> cleaning seem went
exterior sun
47399     Thanksgiving Dinner.  ...  Thanksgiving Dinner. <EOT> house family
though...
47400                  On Sale  ...      On Sale <EOT> searching want saw bought
saved
47401       First Scraped Knee  ...  First Scraped Knee <EOT> eric ground saw
mothe...
47402                       tv  ...          tv <EOT> show writing youtube
started got
```

```
...                          ...  ...
...
52660                    Flavor  ...        Flavor <EOT> flavor tried get recipe
recipe
52661               After Death  ...      After Death <EOT> trouble day found
told week
52662  Janice breaks her wrist  ...  Janice breaks her wrist <EOT> janice legs
work...
52663   Jamie marries for love  ...  Jamie marries for love <EOT> jamie married
man...
52664                    Orange  ...               Orange <EOT> tree hit tree
tree nose

[5267 rows x 4 columns]
```

[ ]:
```python
# Create a dataloading module as per the PyTorch Lightning Docs
SOURCE = 'storytitle'
TARGET = 'storyline'

class SummaryDataModule(pl.LightningDataModule):
  def __init__(self, tokenizer, train, valid, batch_size):
    super().__init__()
    self.tokenizer = tokenizer
    self.train = train
    self.validate = valid
    self.batch_size = batch_size

  # Loads and splits the data into training, validation and test sets with a 80/
↪10/10 split
  # def prepare_data(self):
    # self.data = pd.read_csv(self.data_file)[:self.num_examples]
    # self.train, self.validate, self.test = np.split(self.data.sample(frac=1),␣
↪[int(.8*len(self.data))])

  # encode the sentences using the tokenizer
  def setup(self, stage):
    self.train = encode_sentences(self.tokenizer, self.train[SOURCE], self.
↪train[TARGET])
    self.validate = encode_sentences(self.tokenizer, self.validate[SOURCE],␣
↪self.validate[TARGET])
    # self.test = encode_sentences(self.tokenizer, self.test[SOURCE], self.
↪test[TARGET])

  # Load the training, validation and test sets in Pytorch Dataset objects
  def train_dataloader(self):
    dataset = TensorDataset(self.train['input_ids'], self.
↪train['attention_mask'], self.train['labels'])
```

```
    train_data = DataLoader(dataset, sampler = RandomSampler(dataset),␣
↪batch_size = self.batch_size)
    return train_data

  def val_dataloader(self):
    dataset = TensorDataset(self.validate['input_ids'], self.
↪validate['attention_mask'], self.validate['labels'])
    val_data = DataLoader(dataset, batch_size = self.batch_size)
    return val_data

  # def test_dataloader(self):
  #   dataset = TensorDataset(self.test['input_ids'], self.
↪test['attention_mask'], self.test['labels'])
  #   test_data = DataLoader(dataset, batch_size = self.batch_size)
↪
  #   return test_data
```

[ ]:
```
class LitModel(pl.LightningModule):
  # Instantiate the model
  def __init__(self, learning_rate, tokenizer, model, hparams):
    super().__init__()
    self.tokenizer = tokenizer
    self.model = model
    self.learning_rate = learning_rate
    # self.freeze_encoder = freeze_encoder
    # self.freeze_embeds_ = freeze_embeds
    # self.hparams = hparams
    self.save_hyperparameters(hparams)

    if self.hparams.freeze_encoder:
      freeze_params(self.model.get_encoder())

    if self.hparams.freeze_embeds:
      self.freeze_embeds()

  def freeze_embeds(self):
    ''' freeze the positional embedding parameters of the model; adapted from␣
↪finetune.py '''
    freeze_params(self.model.model.shared)
    for d in [self.model.model.encoder, self.model.model.decoder]:
      freeze_params(d.embed_positions)
      freeze_params(d.embed_tokens)

  # Do a forward pass through the model
  def forward(self, input_ids, **kwargs):
    return self.model(input_ids, **kwargs)
```

```
 def configure_optimizers(self):
   optimizer = torch.optim.Adam(self.parameters(), lr = self.learning_rate)
   return optimizer

 def training_step(self, batch, batch_idx):
   # Load the data into variables
   src_ids, src_mask = batch[0], batch[1]
   tgt_ids = batch[2]
   # Shift the decoder tokens right (but NOT the tgt_ids)
   decoder_input_ids = shift_tokens_right(tgt_ids, tokenizer.pad_token_id)

   # Run the model and get the logits
   outputs = self(src_ids, attention_mask=src_mask,␣
↪decoder_input_ids=decoder_input_ids, use_cache=False)
   lm_logits = outputs[0]
   # Create the loss function
   ce_loss_fct = torch.nn.CrossEntropyLoss(ignore_index=self.tokenizer.
↪pad_token_id)
   # Calculate the loss on the un-shifted tokens
   loss = ce_loss_fct(lm_logits.view(-1, lm_logits.shape[-1]), tgt_ids.
↪view(-1))

   return {'loss':loss}

 def validation_step(self, batch, batch_idx):

   src_ids, src_mask = batch[0], batch[1]
   tgt_ids = batch[2]

   decoder_input_ids = shift_tokens_right(tgt_ids, tokenizer.pad_token_id)

   # Run the model and get the logits
   outputs = self(src_ids, attention_mask=src_mask,␣
↪decoder_input_ids=decoder_input_ids, use_cache=False)
   lm_logits = outputs[0]

   ce_loss_fct = torch.nn.CrossEntropyLoss(ignore_index=self.tokenizer.
↪pad_token_id)
   val_loss = ce_loss_fct(lm_logits.view(-1, lm_logits.shape[-1]), tgt_ids.
↪view(-1))

   # self.log('val_loss', val_loss)

   return {'loss': val_loss}
```

```
 # Method that generates text using the BartForConditionalGeneration's␣
↪generate() method
 def generate_text(self, text, eval_beams, early_stopping = False, max_len =␣
↪5):
   ''' Function to generate text '''
   generated_ids = self.model.generate(
       text["input_ids"],
       attention_mask=text["attention_mask"],
       use_cache=True,
       decoder_start_token_id = self.tokenizer.pad_token_id,
       num_beams= eval_beams,
       max_length = max_len,
       early_stopping = early_stopping
   )
   return [self.tokenizer.decode(w, skip_special_tokens=True,␣
↪clean_up_tokenization_spaces=True) for w in generated_ids]

   # def training_epoch_end(self,outputs):
   #     #  the function is called after every epoch is completed

   #     # calculating average loss
   #     avg_loss = torch.stack([x['loss'] for x in outputs]).mean()

   #     # calculating correect and total predictions
   #     # correct=sum([x["correct"] for  x in outputs])
   #     # total=sum([x["total"] for  x in outputs])

   #     # creating log dictionary
   #     # tensorboard_logs = {'loss': avg_loss,"Accuracy": correct/total}

   #     # epoch_dictionary={
   #     #     # required
   #     #     'loss': avg_loss,

   #     #     # for logging purposes
   #     #     'log': tensorboard_logs}

   #     # return epoch_dictionary
   #     self.log('train_loss', avg_loss)



def freeze_params(model):
  ''' Function that takes a model as input (or part of a model) and freezes the␣
↪layers for faster training
     adapted from finetune.py '''
 for layer in model.parameters():
```

```
    layer.requires_grade = False
```

[ ]:
```
# Create the hparams dictionary to pass in the model
# I realise that this isn't really how this is meant to be used, but having␣
 ↪this here reminds me that I can edit it when I need
hparams = argparse.Namespace()

hparams.freeze_encoder = True
hparams.freeze_embeds = True
hparams.eval_beams = 4
```

[ ]:
```
def shift_tokens_right(input_ids, pad_token_id):
  """ Shift input ids one token to the right, and wrap the last non pad token␣
 ↪(usually <eos>).
      This is taken directly from modeling_bart.py
  """
  prev_output_tokens = input_ids.clone()
  index_of_eos = (input_ids.ne(pad_token_id).sum(dim=1) - 1).unsqueeze(-1)
  prev_output_tokens[:, 0] = input_ids.gather(1, index_of_eos).squeeze()
  prev_output_tokens[:, 1:] = input_ids[:, :-1]
  return prev_output_tokens

def encode_sentences(tokenizer, source_sentences, target_sentences,␣
 ↪max_length=32, pad_to_max_length=True, return_tensors="pt"):
  ''' Function that tokenizes a sentence
      Args: tokenizer - the BART tokenizer; source and target sentences are the␣
 ↪source and target sentences
      Returns: Dictionary with keys: input_ids, attention_mask, target_ids
  '''

  input_ids = []
  attention_masks = []
  target_ids = []
  tokenized_sentences = {}

  for sentence in source_sentences:
    encoded_dict = tokenizer(
          sentence,
          max_length=max_length,
          padding="max_length" if pad_to_max_length else None,
          truncation=True,
          return_tensors=return_tensors,
          add_prefix_space = True
      )

    input_ids.append(encoded_dict['input_ids'])
    attention_masks.append(encoded_dict['attention_mask'])
```

```
  input_ids = torch.cat(input_ids, dim = 0)
  attention_masks = torch.cat(attention_masks, dim = 0)

  for sentence in target_sentences:
    encoded_dict = tokenizer(
          sentence,
          max_length=max_length,
          padding="max_length" if pad_to_max_length else None,
          truncation=True,
          return_tensors=return_tensors,
          add_prefix_space = True
      )
    # Shift the target ids to the right
    # shifted_target_ids = shift_tokens_right(encoded_dict['input_ids'],␣
 ↪tokenizer.pad_token_id)
    target_ids.append(encoded_dict['input_ids'])

  target_ids = torch.cat(target_ids, dim = 0)


  batch = {
      "input_ids": input_ids,
      "attention_mask": attention_masks,
      "labels": target_ids,
  }

  return batch


# def noise_sentence(sentence_, percent_words, replacement_token = "<mask>"):
#   '''
#   Function that noises a sentence by adding <mask> tokens
#   Args: sentence - the sentence to noise
#         percent_words - the percent of words to replace with <mask> tokens;␣
 ↪the number is rounded up using math.ceil
#   Returns a noised sentence
#   '''
#   # Create a list item and copy
#   sentence_ = sentence_.split(' ')
#   sentence = sentence_.copy()

#   num_words = math.ceil(len(sentence) * percent_words)

#   # Create an array of tokens to sample from; don't include the last word as␣
 ↪an option because in the case of lyrics
#   # that word is often a rhyming word and plays an important role in song␣
 ↪construction
```

```
#   sample_tokens = set(np.arange(0, np.maximum(1, len(sentence)-1)))

#   words_to_noise = random.sample(sample_tokens, num_words)

#   # Swap out words, but not full stops
#   for pos in words_to_noise:
#       if sentence[pos] != '.':
#           sentence[pos] = replacement_token

#   # Remove redundant spaces
#   sentence = re.sub(r' {2,5}', ' ', ' '.join(sentence))

#   # Combine concurrent <mask> tokens into a single token; this just does two␣
 ↪rounds of this; more could be done
#   sentence = re.sub(r'<mask> <mask>', "<mask>", sentence)
#   sentence = re.sub(r'<mask> <mask>', "<mask>", sentence)
#   return sentence
```

2 Load BART

Here we load the model. I used "bart-base" because I had memory issues using "bart-large". "bart-base" appears to load without the use_cache argument, which by necessity must be turned to "False" for "bart-large".

[ ]:
```
# Load the model
from transformers import BartTokenizer, BartForConditionalGeneration, AdamW,␣
 ↪BartConfig

tokenizer = BartTokenizer.from_pretrained('facebook/bart-base',␣
 ↪add_prefix_space=True)

bart_model = BartForConditionalGeneration.from_pretrained(
    "facebook/bart-base")
```

[ ]:
```
# Load the data into the model for training
summary_data = SummaryDataModule(tokenizer, train, test, batch_size = 16)

# Load the model from a pre-saved checkpoint; alternatively use the code below␣
 ↪to start training from scratch
# model = LitModel.load_from_checkpoint(root_dir + "checkpoint_files_2/
 ↪10_epoch_11_29_lt.ckpt",
#                                       learning_rate = 2e-5, tokenizer =␣
 ↪tokenizer, model = bart_model, hparams = hparams)

model = LitModel(learning_rate = 2e-5, tokenizer = tokenizer, model =␣
 ↪bart_model, hparams = hparams)
```

3 Training the model with Pytorch Lightning

The below code utilises Pytorch Lightning's fantastic Trainer module that helps to control the training process. After creating a ModelCheckpoint object, the other options are fed into the Trainer module. I found that my colab crashed when I didn't explicitly set progress_bar_refresh_rate to something and I found that setting it to 500 seemed to work just fine.

[ ]:
```
checkpoint = ModelCheckpoint(dirpath=root_dir + 'checkpoint_files_2/')
trainer = pl.Trainer(gpus = 1,
                     max_epochs = 5,
                     min_epochs = 5,
                     auto_lr_find = False,
                     checkpoint_callback = checkpoint,
                     progress_bar_refresh_rate = 500)
```

```
/usr/local/lib/python3.7/dist-
packages/pytorch_lightning/trainer/connectors/callback_connector.py:148:
LightningDeprecationWarning: Setting `Trainer(checkpoint_callback=<pytorch_light
ning.callbacks.model_checkpoint.ModelCheckpoint object at 0x7fa92b82fc10>)` is
deprecated in v1.5 and will be removed in v1.7. Please consider using `Trainer(e
nable_checkpointing=<pytorch_lightning.callbacks.model_checkpoint.ModelCheckpoin
t object at 0x7fa92b82fc10>)`.
  f"Setting `Trainer(checkpoint_callback={checkpoint_callback})` is deprecated
in v1.5 and will "
/usr/local/lib/python3.7/dist-
packages/pytorch_lightning/trainer/connectors/callback_connector.py:91:
LightningDeprecationWarning: Setting `Trainer(progress_bar_refresh_rate=500)` is
deprecated in v1.5 and will be removed in v1.7. Please pass
`pytorch_lightning.callbacks.progress.TQDMProgressBar` with `refresh_rate`
directly to the Trainer's `callbacks` argument instead. Or, to disable the
progress bar pass `enable_progress_bar = False` to the Trainer.
  f"Setting `Trainer(progress_bar_refresh_rate={progress_bar_refresh_rate})` is
deprecated in v1.5 and"
GPU available: True, used: True
TPU available: False, using: 0 TPU cores
IPU available: False, using: 0 IPUs
```

[ ]:
```
# Fit the instantiated model to the data
trainer.fit(model, summary_data)
```

```
LOCAL_RANK: 0 - CUDA_VISIBLE_DEVICES: [0]

  | Name  | Type                          | Params
-------------------------------------------------------
0 | model | BartForConditionalGeneration | 139 M
-------------------------------------------------------
139 M     Trainable params
```

```
0         Non-trainable params
139 M     Total params
557.682   Total estimated model params size (MB)

Validation sanity check: 0it [00:00, ?it/s]


Training: 0it [00:00, ?it/s]


Validating: 0it [00:00, ?it/s]


Validating: 0it [00:00, ?it/s]


Validating: 0it [00:00, ?it/s]


Validating: 0it [00:00, ?it/s]


Validating: 0it [00:00, ?it/s]
```

[1]:
```
# %load_ext tensorboard
# %tensorboard --logdir /content/lightning_logs/
```

[ ]:
```
# If you want to manually save a checkpoint, this works, although the model␣
 ↪should automatically save (progressively better)
# checkpoints as it moves through the epochs
trainer.save_checkpoint(root_dir + "checkpoint_files_2/5_epoch_11_30_lt.ckpt")
```

4 Generate Storylines

[ ]:
```
def generate_storyline(title, model_):
  # Put the model on eval mode
  model_.to(torch.device('cpu'))
  model_.eval()
  prompt_line_tokens = tokenizer(title, max_length = 100, return_tensors =␣
 ↪"pt", truncation = True)
  out = model.generate_text(prompt_line_tokens, eval_beams = 4, early_stopping␣
 ↪= True, max_len=1000)
  return out[0].strip()
```

[ ]:
```
storytitle = df['storytitle'][0]
storytitle
```

[ ]:
```
'David Drops the Weight'
```

[ ]:
```
storyline = df['storyline'][0]
storyline
```

[ ]:
```
'put try realized started weeks'
```

[ ]:
```
storyline = generate_storyline(storytitle, model)
storyline
#3 epoch
#Frustration
#jimmy told told told said
#David Drops the Weight
#dave overweight weight weight weight
#8 epoches
#David Drops the Weight
#lose wanted diet doctor doctor
#Frustration
#jim jim told told jim
#13 epoches
# Frustration
# tom angry wall tom tom
# David Drops the Weight
# pounds decided went month pounds
```

[ ]:
```
'lose pounds pounds pounds happy'
```

[ ]:
```
test
```

[ ]:
```
                    storytitle  ...
titleLineConcat
47398                   Crazed  ...       Crazed <EOT> cleaning seem went
exterior sun
47399     Thanksgiving Dinner.  ...  Thanksgiving Dinner. <EOT> house family
though...
47400                  On Sale  ...      On Sale <EOT> searching want saw bought
saved
47401       First Scraped Knee  ...  First Scraped Knee <EOT> eric ground saw
mothe...
47402                       tv  ...          tv <EOT> show writing youtube
started got
...                        ...  ...
...
52660                   Flavor  ...        Flavor <EOT> flavor tried get recipe
recipe
52661              After Death  ...      After Death <EOT> trouble day found
told week
52662  Janice breaks her wrist  ...  Janice breaks her wrist <EOT> janice legs
work...
52663   Jamie marries for love  ...  Jamie marries for love <EOT> jamie married
man...
```

```
52664                   Orange  ...                Orange <EOT> tree hit tree
tree nose

[5267 rows x 4 columns]
```

[ ]:
```
reduced_test, _= np.split(test, [int(.05*len(test))])
reduced_test
```

[ ]:
```
                storytitle  ...
titleLineConcat
47398                Crazed  ...       Crazed <EOT> cleaning seem went exterior
sun
47399  Thanksgiving Dinner.  ...  Thanksgiving Dinner. <EOT> house family
though...
47400               On Sale  ...      On Sale <EOT> searching want saw bought
saved
47401    First Scraped Knee  ...  First Scraped Knee <EOT> eric ground saw
mothe...
47402                    tv  ...          tv <EOT> show writing youtube started
got
...                     ...  ...
...
47656            Rear Ended  ...     Rear Ended <EOT> police man site trying
subdue
47657                   Bat  ...           Bat <EOT> baseball made swung half
larry
47658              Her Mom  ...  Her Mom <EOT> estranged seen contacts
reacquai...
47659  Pat the Photographer  ...  Pat the Photographer <EOT> park saw took
photo...
47660              Pressure  ...  Pressure <EOT> bought decided water panicked
s...

[263 rows x 4 columns]
```

[ ]:
```
remove_index = []
generated_lines = []
for index, row in reduced_test.iterrows():
    out = generate_storyline(row['storytitle'], model)
    if len(out.split()) != 5 or '.' in out.split():
      remove_index.append(index)
      print(out)
    else:
      generated_lines.append(out)
```

```
gas station gas station station gas
girl scout troop troop scout troop
julie birthday jill jill party jill
cookout tryouts tryouts success
```

```
rescuers rescuers rescued rescuers
wendy wagon wagon wagon truck wagon
```

[ ]:
```
reduced_test.drop(remove_index, axis=0, inplace=True)
```

[ ]:
```
reduced_test['predicted_storylines'] = generated_lines
```

[ ]:
```
for index, row in reduced_test.iterrows():
    out = row['predicted_storylines']
    if len(out.split()) != 5 or '.' in out.split():
      print(out)
```

[ ]:
```
reduced_test
```

[ ]:
```
                storytitle  ...                      predicted_storylines
47398               Crazed  ...                kate school kate told kate
47399  Thanksgiving Dinner.  ...  thanksgiving family turkey turkey turkey
47400               On Sale  ...               sale wanted wanted went sale
47401    First Scraped Knee  ...                    tom knee tom wound tom
47402                    tv  ...                         tv watch tv tv tv
...                     ...  ...                                      ...
47656           Rear Ended  ...                      tom side tom tom tom
47657                  Bat  ...                       bat bat hit hit bat
47658              Her Mom  ...           visit visit visit visited visit
47659  Pat the Photographer  ...  patrick photographer patrick shot patrick
47660             Pressure  ...                jane wanted went went jane

[257 rows x 5 columns]
```

[ ]:
```
def concat_title_line(x):
    l = x['storytitle'].split()
    l.append('<EOT>')
    l.extend(x['predicted_storylines'].split())
    return ' '.join(l)

reduced_test['predicted_titleLineConcat'] = reduced_test.apply(lambda x:␣
 ↪concat_title_line(x), axis=1)
```

[ ]:
```
reduced_test
```

[ ]:
```
                storytitle  ...
predicted_titleLineConcat
47398               Crazed  ...           Crazed <EOT> kate school kate told
kate
47399  Thanksgiving Dinner.  ...  Thanksgiving Dinner. <EOT> thanksgiving
family...
47400               On Sale  ...         On Sale <EOT> sale wanted wanted went
sale
47401    First Scraped Knee  ...    First Scraped Knee <EOT> tom knee tom wound
tom
47402                    tv  ...                         tv <EOT> tv watch tv tv
```

```
tv
...                     ...  ...
...
47656            Rear Ended  ...              Rear Ended <EOT> tom side tom tom
tom
47657                   Bat  ...                     Bat <EOT> bat bat hit hit
bat
47658              Her Mom  ...      Her Mom <EOT> visit visit visit visited
visit
47659  Pat the Photographer  ...  Pat the Photographer <EOT> patrick
photographe...
47660              Pressure  ...          Pressure <EOT> jane wanted went went
jane

[257 rows x 6 columns]
```

[ ]:
```
reduced_test.to_csv(root_dir+'test_predicted_storyline_5_ep.csv',␣
↪encoding='utf-8', index=False)
```

[ ]:
```
#output notebook as pdf
!wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf('bart_title_2_storyline.ipynb')
```

```
--2021-12-01 05:18:29--  https://raw.githubusercontent.com/brpy/colab-
pdf/master/colab_pdf.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)...
185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com
(raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1864 (1.8K) [text/plain]
Saving to: colab_pdf.py

colab_pdf.py        100%[===================>]   1.82K  --.-KB/s    in 0s

2021-12-01 05:18:30 (29.9 MB/s) - colab_pdf.py saved [1864/1864]

Mounted at /content/drive/

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.


WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Extracting templates from packages: 100%
```