



# Deep Learning School

**Физтех-Школа Прикладной математики и информатики (ФПМИ) МФТИ**

## Text Summarization

Всем привет! Сегодня мы познакомимся с задачей суммаризации текста на примере генерации "сжатых" новостей. Рассмотрим некоторые базовые решения и познакомимся с архитектурами нейросетей для решения задачи. Датасет: gazeta.ru

Ноутбук создан на основе семинара Гусева Ильи на кафедре компьютерной лингвистики АВВУУ МФТИ.

Загрузим датасет и необходимые библиотеки

```
In [1]: from google.colab import drive
```

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

Mounted at /content/drive

```
In [2]: import os
os.chdir("/content/drive/MyDrive/Colab_Notebooks/DLSchool_mipt_2sem")
```

```
In [3]: # !wget -q https://www.dropbox.com/s/43l702z5a5i2w8j/gazeta_train.txt
# !wget -q https://www.dropbox.com/s/k2egt3sug0hb185/gazeta_val.txt
# !wget -q https://www.dropbox.com/s/3gki5n5djs9w0v6/gazeta_test.txt
```

```
In [4]: !pip -q install razdel networkx pymorphy2 nltk rouge==0.3.1 summa
```

```
61kB 3.9MB/s
61kB 6.7MB/s
8.2MB 10.6MB/s
Building wheel for summa (setup.py) ... done
```

## SEED

```
In [5]: import torch, random
import numpy as np

SEED = 42

random.seed(SEED)
np.random.seed(SEED)
torch.manual_seed(SEED)
torch.cuda.manual_seed(SEED)
torch.backends.cudnn.deterministic = True
```

## Dataset

Посмотрим на то, как устроен датасет

```
In [6]: !head -n 1 gazeta_train.txt
# with open("gazeta_train.txt", encoding="utf-8") as f:
#     print(f.readline())
!cat gazeta_train.txt | wc -l
!cat gazeta_val.txt | wc -l
!cat gazeta_test.txt | wc -l
```

```
{"url": "https://www.gazeta.ru/financial/2011/11/30/3852658.shtml", "text": "«По итогам 2011 года чистый отток может составить при
```

мерно \$80 млрд, в следующем году – около \$20 млрд. При этом мы ожидаем, что со второго полугодия 2012 года начнется приток капитала», – заявил «Интерфаксу» замминистра экономического развития Андрей Клепач. Официальные прогнозы по выводу капитала из России становятся все пессимистичными: еще летом власти полагали, что из страны уйдет не более \$35 млрд, в сентябре Минэкономразвития назвал цифру \$50 млрд, в начале ноября Центробанк пересмотрел оценку до \$70 млрд. Очередное изменение прогноза было ожидаемо: по расчетам Центробанка, за январь – октябрь чистый отток капитала достиг \$64 млрд, причем в последние месяцы он ускорился: в сентябре он составил \$14 млрд, в октябре – \$13 млрд против среднего ежемесячного оттока в \$6–8 млрд в первом полугодии. «После октябрьских данных Минэкономразвития вынуждено было изменить оценку, настаивать на \$70 млрд означало ожидать серьезного замедления оттока капитала на непонятно каких причинах», – говорит главный экономист BNP Paribas Юлия Цепляева. «В последние два месяца отток капитала ускорился, на декабрь приходится значительная часть выплат по внешним долгам, что приводит к усилению оттока, особенно если они не рефинансируются новыми кредитами», – соглашается главный экономист ФК «Открытие» Владимир Тихомиров. Прогнозируемый Минэкономразвития отток капитала – один из самых высоких за последние 20 лет. Больше ушло лишь в 2008 году на фоне разрастания финансового кризиса и российско-грузинской войны – \$133,7 млрд. В кризисный 2009 год из России утекло \$56,1 млрд. Главный фактор ускорения оттока капитала в 2011 году – нестабильность на внешних финансовых рынках и рост опасений относительно второй волны рецессии. «Это реакция на неувренность, которую генерирует Европа с долговыми проблемами. В случае новой волны глобальной турбулентности Россия – одна из самых уязвимых стран», – говорит Цепляева. Еще одна причина – ослабление рубля. «Привлекательность вложений снижается на фоне того, что рубль перестал укрепляться, а ставки по депозитам достаточно низкие. В результате экспортеры не полностью возвращают экспортную выручку», – говорит Тихомиров. Внутри страны эксперты не видят особых причин для бегства капитала. «Ситуация выглядит достаточно позитивно, очень хорошие макроэкономические результаты за год, особенно на фоне других стран. С политической точки зрения все достаточно понятно и предсказуемо, итог выборов очевиден», – говорит экономист ИК «Тройка Диалог» Антон Струченевский. Тем не менее политический фактор играет роль. «Бизнесу важно не только, кто будет президентом, он ждет ясности с перестановками в правительстве. В наших условиях административный ресурс важнее всего для успешности бизнеса», – говорит Цепляева, добавляя, что отток капитала продолжится до завершения президентских выборов.", "title": "Прогноз не успевает за оттоком", "summary": "В 2011 году из России уйдет \$80 млрд, считают в Минэкономразвития. Менее месяца назад Центробанк давал оценку \$70 млрд, повысив первоначальный прогноз вдвое. Отток капитала из страны усиливается из-за кризиса в Европе, а в декабре российским компаниям выплачивать внешние долги. На движение капитала повлияли и выборы: несмотря на их предсказуемость, бизнес хочет ясности с перестановками в правительстве.", "date": "2011-11-30 18:33:39"}

52400

5265

5770

In [7]:

```
import json
import random

def read_gazeta_records(file_name, shuffle=True, sort_by_date=False):
    assert shuffle != sort_by_date
    records = []
    with open(file_name, "r") as r:
        for line in r:
            records.append(json.loads(line))
    if sort_by_date:
        records.sort(key=lambda x: x["date"])
    if shuffle:
        random.shuffle(records)
    return records
```

In [8]:

```
train_records = read_gazeta_records("gazeta_train.txt")
```

```
val_records = read_gazeta_records("gazeta_val.txt")
test_records = read_gazeta_records("gazeta_test.txt")
```

```
In [9]: from nltk.translate.bleu_score import corpus_bleu
        from rouge import Rouge

        def calc_scores(references, predictions, metric="all"):
            print("Count:", len(predictions))
            print("Ref:", references[-1])
            print("Hyp:", predictions[-1])

            if metric in ("bleu", "all"):
                print("BLEU: ", corpus_bleu([[r] for r in references], predictions))
            if metric in ("rouge", "all"):
                try:
                    rouge = Rouge()
                    scores = rouge.get_scores(predictions, references, avg=True)
                    print("ROUGE: ", scores)
                except ValueError as err:
                    print(f"Error by ROUGE score calculation: {err}")
```

## Extractive RNN

### BPE

Для начала сделаем BPE токенизацию

```
In [10]: !pip install youtokentome
```

```
Collecting youtokentome
  Downloading https://files.pythonhosted.org/packages/c8/1c/224cdc3d9a32ed706c8fb1f30b491be6ea5da114ff4edc174014cc24fa43/youtokentome-1.0.6-cp37-cp37m-manylinux2010_x86_64.whl (1.7MB)
    |████████████████████████████████████████| 1.7MB 5.1MB/s
Requirement already satisfied: Click>=7.0 in /usr/local/lib/python3.7/dist-packages (from youtokentome) (7.1.2)
Installing collected packages: youtokentome
Successfully installed youtokentome-1.0.6
```

```
In [11]: %%time
        import youtokentome as yttm

        def train_bpe(records, model_path, model_type="bpe", vocab_size=10000, lower=True):
            temp_file_name = "temp.txt"
```

```

with open(temp_file_name, "w") as temp:
    for record in records:
        text, summary = record['text'], record['summary']
        if lower:
            summary = summary.lower()
            text = text.lower()
        if not text or not summary:
            continue
        temp.write(text + "\n")
        temp.write(summary + "\n")
yttm.BPE.train(data=temp_file_name, vocab_size=vocab_size, model=model_path)

train_bpe(train_records, "BPE_model.bin")

```

CPU times: user 27.5 s, sys: 2.29 s, total: 29.8 s  
 Wall time: 24.4 s

## Словарь

Составим словарь для индексации токенов

```

In [12]: bpe_processor = yttm.BPE('BPE_model.bin')
         vocabulary = bpe_processor.vocab()

```

## Кэш oracle summary

Закешируем oracle summary, чтобы не пересчитывать их каждый раз

```

In [13]: from rouge import Rouge
         import razdel
         from tqdm.notebook import tqdm

         import copy

         def build_oracle_summary_greedy(text, gold_summary, calc_score, lower=True, max_sentences=30):
             """
             Жадное построение oracle summary
             """

             gold_summary = gold_summary.lower() if lower else gold_summary
             # Делим текст на предложения
             sentences = [sentence.text.lower() if lower else sentence.text for sentence in razdel.sentenzize(text)][:max_sentences]
             n_sentences = len(sentences)
             oracle_summary_sentences = set()

```

```

score = -1.0
summaries = []
for _ in range(n_sentences):
    for i in range(n_sentences):
        if i in oracle_summary_sentences:
            continue
        current_summary_sentences = copy.copy(oracle_summary_sentences)
        # Добавляем какое-то предложения к уже существующему summary
        current_summary_sentences.add(i)
        current_summary = " ".join([sentences[index] for index in sorted(list(current_summary_sentences))])
        # Считаем метрики
        current_score = calc_score(current_summary, gold_summary)
        summaries.append((current_score, current_summary_sentences))
        # Если получилось улучшить метрики с добавлением какого-либо предложения, то пробуем добавить ещё
        # Иначе на этом заканчиваем
        best_summary_score, best_summary_sentences = max(summaries)
        if best_summary_score <= score:
            break
        oracle_summary_sentences = best_summary_sentences
        score = best_summary_score
oracle_summary = " ".join([sentences[index] for index in sorted(list(oracle_summary_sentences))])
return oracle_summary, oracle_summary_sentences

def calc_single_score(pred_summary, gold_summary, rouge):
    return rouge.get_scores([pred_summary], [gold_summary], avg=True)['rouge-2']['f']

def add_oracle_summary_to_records(records, max_sentences=30, lower=True, nrows=1000):
    rouge = Rouge()
    for i, record in tqdm(enumerate(records), total=nrows):
        if i >= nrows:
            break
        text = record["text"]
        summary = record["summary"]

        summary = summary.lower() if lower else summary
        sentences = [sentence.text.lower() if lower else sentence.text for sentence in razdel.sentence(text)][0:max_sentences]
        oracle_summary, sentences_indicies = build_oracle_summary_greedy(text, summary, calc_score=lambda x, y: calc_single_score(
            lower=lower, max_sentences=max_sentences)

        record["sentences"] = sentences
        record["oracle_sentences"] = list(sentences_indicies)
        record["oracle_summary"] = oracle_summary

    return records[:nrows]

```

```
ext_train_records = add_oracle_summary_to_records(train_records, nrows=2048)
ext_val_records = add_oracle_summary_to_records(val_records, nrows=256)
ext_test_records = add_oracle_summary_to_records(test_records, nrows=256)
```

## Составление батчей

```
In [14]: import torch
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
device
```

```
Out[14]: device(type='cuda')
```

```
In [15]: import random
import math
import razdel
import torch
import numpy as np
from rouge import Rouge

class BatchIterator():
    def __init__(self, records, vocabulary, batch_size, bpe_processor, shuffle=True, lower=True, max_sentences=30, max_sentence_length=100):
        self.records = records
        self.num_samples = len(records)
        self.batch_size = batch_size
        self.bpe_processor = bpe_processor
        self.shuffle = shuffle
        self.batches_count = int(math.ceil(self.num_samples / batch_size))
        self.lower = lower
        self.rouge = Rouge()
        self.vocabulary = vocabulary
        self.max_sentences = max_sentences
        self.max_sentence_length = max_sentence_length
        self.device = device

    def __len__(self):
        return self.batches_count

    def __iter__(self):
```

```
indices = np.arange(self.num_samples)
if self.shuffle:
    np.random.shuffle(indices)

for start in range(0, self.num_samples, self.batch_size):
    end = min(start + self.batch_size, self.num_samples)
    batch_indices = indices[start:end]

    batch_inputs = []
    batch_outputs = []
    max_sentence_length = 0
    max_sentences = 0
    batch_records = []

    for data_ind in batch_indices:

        record = self.records[data_ind]
        batch_records.append(record)
        text = record["text"]
        summary = record["summary"]
        summary = summary.lower() if self.lower else summary

        if "sentences" not in record:
            sentences = [sentence.text.lower() if self.lower else sentence.text for sentence in razdel.sentenize(text)][:s
        else:
            sentences = record["sentences"]
        max_sentences = max(len(sentences), max_sentences)

        # номера предложений, которые в нашем саммари
        if "oracle_sentences" not in record:
            calc_score = lambda x, y: calc_single_score(x, y, self.rouge)
            sentences_indicies = build_oracle_summary_greedy(text, summary, calc_score=calc_score, lower=self.lower, max_s
        else:
            sentences_indicies = record["oracle_sentences"]

        # inputs - индексы слов в предложении
        inputs = [bpe_processor.encode(sentence)[:self.max_sentence_length] for sentence in sentences]
        max_sentence_length = max(max_sentence_length, max([len(tokens) for tokens in inputs]))

        # получение метки класса предложения
        outputs = [int(i in sentences_indicies) for i in range(len(sentences))]
        batch_inputs.append(inputs)
        batch_outputs.append(outputs)
```



```

tensor_inputs = torch.zeros((self.batch_size, max_sentences, max_sentence_length), dtype=torch.long, device=self.device)
# we add index 2 for padding
# YOUR CODE
tensor_outputs = torch.ones((self.batch_size, max_sentences), dtype=torch.float32, device=self.device) * 2
# tensor_outputs = torch.zeros((self.batch_size, max_sentences), dtype=torch.float32, device=self.device)

for i, inputs in enumerate(batch_inputs):
    for j, sentence_tokens in enumerate(inputs):
        tensor_inputs[i][j][:len(sentence_tokens)] = torch.tensor(sentence_tokens, dtype=torch.int64)

for i, outputs in enumerate(batch_outputs):
    tensor_outputs[i][:len(outputs)] = torch.LongTensor(outputs)

tensor_outputs = tensor_outputs.long()
yield {
    'inputs': tensor_inputs,
    'outputs': tensor_outputs,
    'records': batch_records
}

```

```

In [16]: train_iterator = BatchIterator(ext_train_records, vocabulary, 32, bpe_processor, device=device)
        val_iterator = BatchIterator(ext_val_records, vocabulary, 32, bpe_processor, device=device)
        test_iterator = BatchIterator(ext_test_records, vocabulary, 32, bpe_processor, device=device)

```

## Extractor - SummaRuNNer

<https://arxiv.org/pdf/1611.04230.pdf>

## Homework

- В данной реализации в `outputs` в качестве padding используется индекс 0. Измените в функции `__iter__` индекс padding, чтобы он не совпадал с классом 0 или 1, например, 2.
- В качестве criterion используйте `CrossEntropyLoss` вместо `BCEWithLogitsLoss`
- Из-за смены criterion, вы уже должны подавать на вход criterion ни одно число, а logits для каждого класса. Перед подачей logits вы можете отфильтровать предсказания для класса padding. В этом пункте вам придется изменять файл `train_model.py`, а именно функции `train` и `evaluate`.
- Используйте два варианта обучения: с весами в `CrossEntropyLoss` и без

- Также сравните `inference`, когда вы ранжируете logits, и когда вы выбираете предложения, у которых `logits > 0`, в двух вариантах обучения.
- Реализуйте дополнительно характеристику предложения `novelty`. Как влияет добавление `novelty` на качество `summary`?
- Постарайтесь улучшить качество модели, полученной на семинаре:  $\$BLEU \approx 0.45\$$

```
In [17]: import numpy as np

import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
from torch.autograd import Variable

from torch.nn.utils.rnn import pack_padded_sequence as pack
from torch.nn.utils.rnn import pad_packed_sequence as unpack

class SentenceEncoderRNN(nn.Module):
    def __init__(self, input_size, embedding_dim, hidden_size, n_layers=3, dropout=0.3, bidirectional=True):
        super().__init__()

        num_directions = 2 if bidirectional else 1
        assert hidden_size % num_directions == 0
        hidden_size = hidden_size // num_directions

        self.embedding_dim = embedding_dim
        self.input_size = input_size
        self.hidden_size = hidden_size
        self.n_layers = n_layers
        self.dropout = dropout
        self.bidirectional = bidirectional

        self.embedding_layer = nn.Embedding(input_size, embedding_dim)
        self.rnn_layer = nn.LSTM(embedding_dim, hidden_size, n_layers, dropout=dropout, bidirectional=bidirectional, batch_first=True)
        self.dropout_layer = nn.Dropout(dropout)

    def forward(self, inputs, hidden=None):
        embedded = self.dropout_layer(self.embedding_layer(inputs))
        outputs, _ = self.rnn_layer(embedded, hidden)
        sentences_embeddings = torch.mean(outputs, 1)
        # [batch_size, hidden_size]
        return sentences_embeddings

class SentenceTaggerRNN(nn.Module):
```

```
def __init__(self,
              vocabulary_size,
              use_content=True,
              use_salience=True,
              use_novelty=True,
              token_embedding_dim=128,
              sentence_encoder_hidden_size=256,
              hidden_size=256,
              bidirectional=True,
              sentence_encoder_n_layers=2,
              sentence_encoder_dropout=0.3,
              sentence_encoder_bidirectional=True,
              n_layers=2,
              dropout=0.3):

    super().__init__()

    num_directions = 2 if bidirectional else 1
    assert hidden_size % num_directions == 0
    hidden_size = hidden_size // num_directions

    self.hidden_size = hidden_size
    self.n_layers = n_layers
    self.dropout = dropout
    self.bidirectional = bidirectional

    self.sentence_encoder = SentenceEncoderRNN(vocabulary_size, token_embedding_dim,
                                              sentence_encoder_hidden_size, sentence_encoder_n_layers,
                                              sentence_encoder_dropout, sentence_encoder_bidirectional)

    self.rnn_layer = nn.LSTM(sentence_encoder_hidden_size, hidden_size, n_layers, dropout=dropout,
                             bidirectional=bidirectional, batch_first=True)

    self.dropout_layer = nn.Dropout(dropout)
    self.content_linear_layer = nn.Linear(hidden_size * 2, 1)
    self.document_linear_layer = nn.Linear(hidden_size * 2, hidden_size * 2)
    self.salience_linear_layer = nn.Linear(hidden_size * 2, hidden_size * 2)
    if use_novelty:
        self.novelty_linear_layer = nn.Linear(hidden_size * 2, hidden_size * 2) # YOUR CODE
    self.tanh_layer = nn.Tanh()

    self.use_content = use_content
    self.use_salience = use_salience
    self.use_novelty = use_novelty
```

```

def forward(self, inputs, hidden=None):
    # parameters of the probability
    content = 0
    salience = 0
    novelty = 0

    # [batch_size, seq num, seq_len]
    batch_size = inputs.size(0)
    sentences_count = inputs.size(1)
    tokens_count = inputs.size(2)
    inputs = inputs.reshape(-1, tokens_count)
    # [batch_size * seq num, seq_len]

    embedded_sentences = self.sentence_encoder(inputs)
    embedded_sentences = self.dropout_layer(embedded_sentences.reshape(batch_size, sentences_count, -1))
    # [batch_size * seq num, seq_len, hidden_size] -> [batch_size, seq num, hidden_size]

    outputs, _ = self.rnn_layer(embedded_sentences, hidden)
    # [batch_size, seq num, 2 * hidden_size] MM

    document_embedding = self.tanh_layer(self.document_linear_layer(torch.mean(outputs, 1)))
    # [batch_size, 2 * hidden_size] MM

    # W * h^T
    if self.use_content:
        content = self.content_linear_layer(outputs).squeeze(2) # 1-representation
        # [batch_size, seq num]

    # h^T * W * d
    if self.use_salience:
        salience = torch.bmm(outputs, self.salience_linear_layer(document_embedding).unsqueeze(2)).squeeze(2) # 2-representation
        # [batch_size, seq num, hidden_size] * [batch_size, hidden_size, 1] = [batch_size, seq num]

    out = content + salience

    if self.use_novelty:
        # at every step add novelty to prediction of the sentence
        predictions = content + salience

        # 0) initialize summary_representation and novelty by zeros
        # YOUR CODE
        summary_representation = torch.zeros_like(outputs[:, 0, :])
        novelty = torch.zeros_like(outputs[:, 0, 0])

```

```

for sentence_num in range(sentences_count):

    # 1) take sentence_num_state from outputs(representation of the sentence with number sentence_num)
    # 2) calculate novelty for current sentence
    # 3) add novelty to predictions
    # 4) calculate probability for current sentence
    # 5) add sentence_num_state with the weight which is equal to probability to summary_representation

    # YOUR CODE
    sentence_num_state = outputs[:, sentence_num, :]
    novelty = torch.bmm(sentence_num_state.unsqueeze(1),
                        self.novelty_linear_layer(torch.tanh(summary_representation))
                        .unsqueeze(2)).squeeze(2).squeeze(1)
    predictions[:, sentence_num] -= novelty
    probability = torch.sigmoid(predictions[:, sentence_num]).unsqueeze(1)
    summary_representation += probability * sentence_num_state

out = predictions

out = torch.cat((-out.unsqueeze(-1), out.unsqueeze(-1)), dim=-1)
return out

```

## Model

$$P(y_j = 1 | \mathbf{h}_j, \mathbf{s}_j, \mathbf{d}) = \sigma(W_c \mathbf{h}_j + \mathbf{h}_j^T W_s)$$

```

In [18]: vocab_size = len(vocabulary)
         model = SentenceTaggerRNN(vocab_size, use_novelty=False).to(device)

         params_count = np.sum([p.numel() for p in model.parameters() if p.requires_grad])
         print("Trainable params: {}".format(params_count))

```

Trainable params: 2861825

```

In [19]: for name, param in model.named_parameters():
         print(f"{name}: {param.numel()}")

```

sentence\_encoder.embedding\_layer.weight: 1280000

```
sentence_encoder.rnn_layer.weight_ih_l0: 65536
sentence_encoder.rnn_layer.weight_hh_l0: 65536
sentence_encoder.rnn_layer.bias_ih_l0: 512
sentence_encoder.rnn_layer.bias_hh_l0: 512
sentence_encoder.rnn_layer.weight_ih_l0_reverse: 65536
sentence_encoder.rnn_layer.weight_hh_l0_reverse: 65536
sentence_encoder.rnn_layer.bias_ih_l0_reverse: 512
sentence_encoder.rnn_layer.bias_hh_l0_reverse: 512
sentence_encoder.rnn_layer.weight_ih_l1: 131072
sentence_encoder.rnn_layer.weight_hh_l1: 65536
sentence_encoder.rnn_layer.bias_ih_l1: 512
sentence_encoder.rnn_layer.bias_hh_l1: 512
sentence_encoder.rnn_layer.weight_ih_l1_reverse: 131072
sentence_encoder.rnn_layer.weight_hh_l1_reverse: 65536
sentence_encoder.rnn_layer.bias_ih_l1_reverse: 512
sentence_encoder.rnn_layer.bias_hh_l1_reverse: 512
rnn_layer.weight_ih_l0: 131072
rnn_layer.weight_hh_l0: 65536
rnn_layer.bias_ih_l0: 512
rnn_layer.bias_hh_l0: 512
rnn_layer.weight_ih_l0_reverse: 131072
rnn_layer.weight_hh_l0_reverse: 65536
rnn_layer.bias_ih_l0_reverse: 512
rnn_layer.bias_hh_l0_reverse: 512
rnn_layer.weight_ih_l1: 131072
rnn_layer.weight_hh_l1: 65536
rnn_layer.bias_ih_l1: 512
rnn_layer.bias_hh_l1: 512
rnn_layer.weight_ih_l1_reverse: 131072
rnn_layer.weight_hh_l1_reverse: 65536
rnn_layer.bias_ih_l1_reverse: 512
rnn_layer.bias_hh_l1_reverse: 512
content_linear_layer.weight: 256
content_linear_layer.bias: 1
document_linear_layer.weight: 65536
document_linear_layer.bias: 256
salience_linear_layer.weight: 65536
salience_linear_layer.bias: 256
```

```
In [20]: # !gdown https://drive.google.com/uc?id=1MiS_iczALcyF7zGDPY6niyeD82P0_PBH -O train_model.py
```

```
In [21]: import train_model
import imp
imp.reload(train_model)
from train_model import train_with_logs
```

## weights

```
In [22]: def get_weights(records):
    one_class_num = 0
    total_class_num = 0
    for item in records:
        one_class_num += len(item["oracle_sentences"])
        total_class_num += len(item["sentences"])
    # weights = torch.tensor([one_class_num / total_class_num, (total_class_num - one_class_num) / total_class_num])
    inv_zero = 1 / (total_class_num - one_class_num)
    inv_one = 1 / one_class_num
    weights = torch.tensor([inv_zero / (inv_zero + inv_one), inv_one / (inv_zero + inv_one)])
    return weights

get_weights(ext_train_records)
```

Out[22]: tensor([0.0790, 0.9210])

```
In [23]: def train(use_class_weights, N_EPOCHS, CLIP, lr=1e-3):
    optimizer = optim.Adam(model.parameters(), lr)
    if use_class_weights:
        # weights depend on the number of objects of class 0 and 1
        # YOUR CODE
        weights = get_weights(ext_train_records).to(device)
        criterion = nn.CrossEntropyLoss(weight=weights)
    else:
        criterion = nn.CrossEntropyLoss()
    train_with_logs(model, train_iterator, val_iterator, optimizer, criterion, N_EPOCHS, CLIP)

# train(True, N_EPOCHS, CLIP)
```

## Inference function

```
In [24]: from train_model import punct_detokenize, postprocess

def inference_summarunner(model, iterator, top_k=3):

    references = []
    predictions = []

    model.eval()
    for batch in test_iterator:
```

```

sum_in = None
logits = model(batch['inputs'][:, :, 1])

if top_k:
    sum_in = torch.argsort(logits, dim=1)[:, -top_k:]

for i in range(len(batch['outputs'])):

    summary = batch['records'][i]['summary'].lower()
    if top_k:
        pred_summary = ' '.join([batch['records'][i]['sentences'][ind] for ind in sum_in.sort(dim=1)[0][i]
                                   if ind < len(batch['records'][i]['sentences'])])
    else:
        pred_summary = ' '.join([batch['records'][i]['sentences'][ind] for ind, logit in enumerate(logits[i])
                                   if (ind < len(batch['records'][i]['sentences'])) and (logit > 0)])

    summary, pred_summary = postprocess(summary, pred_summary)

    references.append(summary)
    predictions.append(pred_summary)

calc_scores(references, predictions)

# model.load_state_dict(torch.load('best-val-model.pt'))
# inference_summarunner(model, test_iterator, 3)

```

## Research Part

### Params init

```

In [25]: N_EPOCHS = 5
         CLIP = 1

```

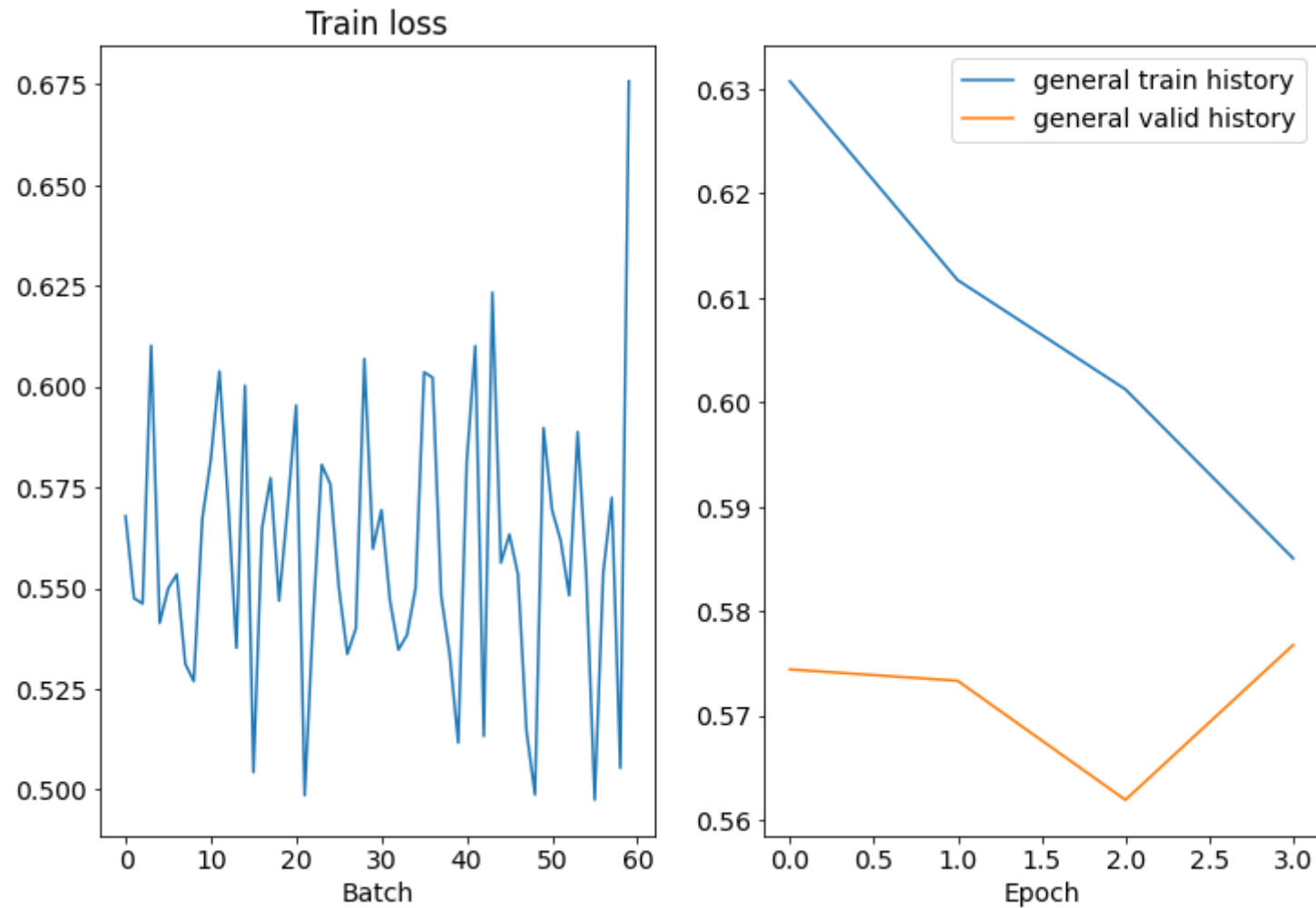
### weights +, novelty +

```

In [26]: model = SentenceTaggerRNN(vocab_size, use_novelty=True).to(device)
         train(use_class_weights=True, N_EPOCHS=N_EPOCHS, CLIP=CLIP)

```





Epoch: 05 | Time: 0m 9s

Train Loss: 0.558 | Train PPL: 1.748

Val. Loss: 0.567 | Val. PPL: 1.763

### top3

```
In [27]: model.load_state_dict(torch.load('best-val-model.pt'))
         inference_summarunner(model, test_iterator, top_k=3)
```

Count: 256

Ref: экс-президент египта хосни мубарак , находившийся у власти целых 30 лет – вплоть до восстания 2011 года , скончался в каире . три десятка лет чрезвычайного положения принесли арабской стране немало изменений , укрепив ее статус на международной арене , но также во многом оказались непростыми для населения .

Нур: бывший президент египта хосни мубарак скончался во вторник в каире , пишут египетские сми . однако официальной информации о смерти 91-летнего экс-президента пока нет . хосни мубарак управлял страной почти 30 лет – с 1981 по 2011 год .

BLEU: 0.4460534146270132

ROUGE: {'rouge-1': {'f': 0.3140955792106141, 'p': 0.29381504725708146, 'r': 0.3551589835085784}, 'rouge-2': {'f': 0.13807773616322402, 'p': 0.12710848739881164, 'r': 0.16146717561073565}, 'rouge-l': {'f': 0.2667689473124119, 'p': 0.26324046454661537, 'r': 0.31712040141618264}}

logits > 0

```
In [28]: inference_summarunner(model, test_iterator, top_k=None)
```

Count: 256

Ref: министр иностранных дел германии хайко маас выступил против вывода ядерного оружия сша с территории фрг . по мнению министра , перемещение американских бомб не решит проблему , так как ядерное оружие следует убрать отовсюду . мир без ядерных вооружений должен оставаться главной целью , подчеркнул маас .

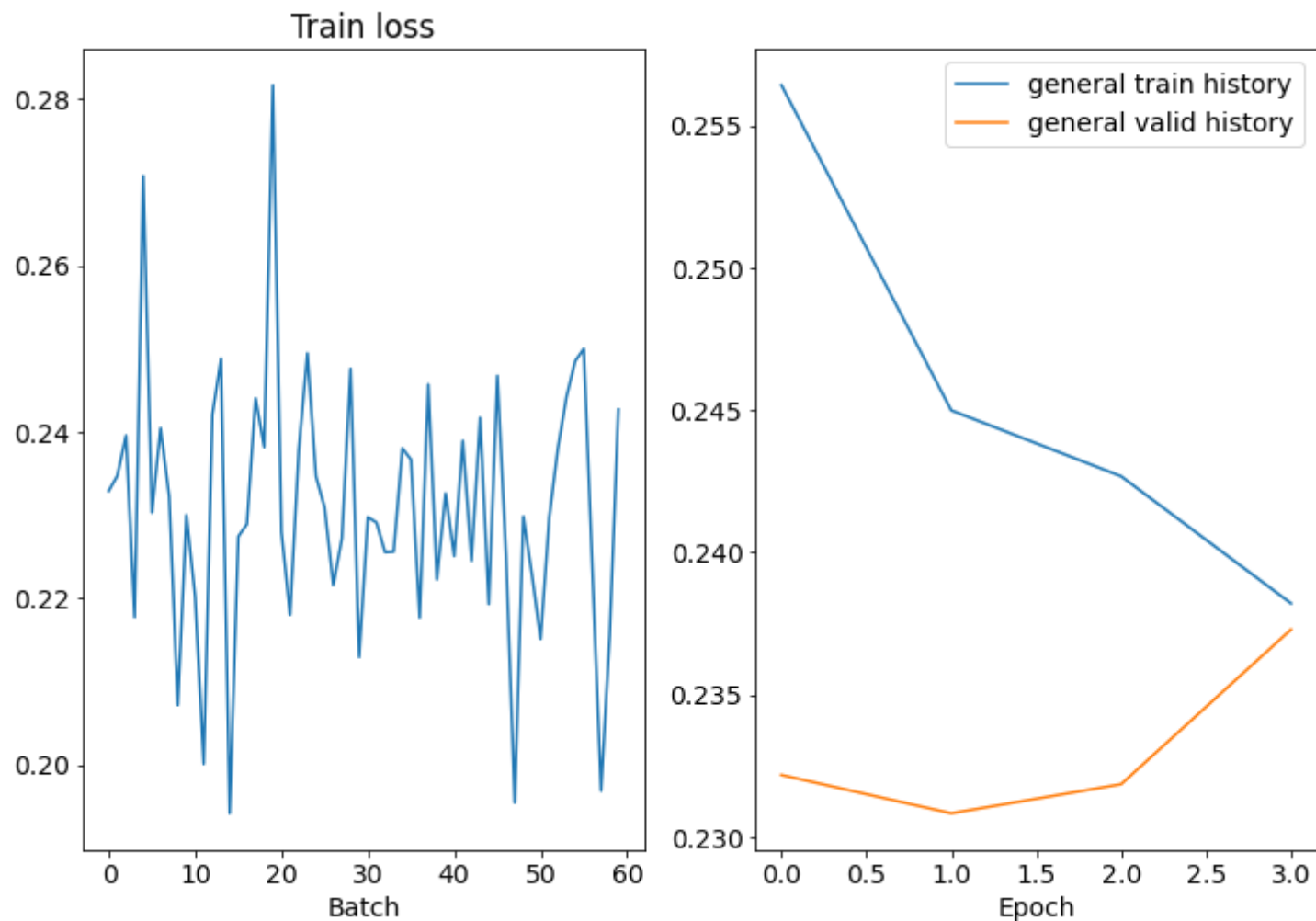
Нур: глава мид германии хайко маас выступил против вывода « в одностороннем порядке » американского ядерного оружия с немецкой территории , призвав запретить данный тип вооружений в целом . « это ничему не поможет , если ядерное оружие переместить из одной страны в другую . если оно должно исчезнуть , оно должно исчезнуть всюду » , – цитирует слова мааса дра . в вопросе ядерного разоружения требуются договоренности на широком базисе , уверен министр , а не в отношении отдельных стран . « нашей целью должен оставаться мир без ядерного оружия , даже если этого непросто достичь или даже если это потребует много времени » , – подчеркнул маас . стоит отметить , что на территории европы находится около 150 единиц тактического ядерного оружия сша , американские вооружения размещены в бельгии , германии , италии и других странах . такие данные приводят сми , но официальной информации на эту тему нет . в частности , газета de morgen летом сообщала , что в фрг хранится около 20 тактических ядерных бомб сша .

BLEU: 0.2993153050453926

ROUGE: {'rouge-1': {'f': 0.2962869129653551, 'p': 0.2291196181518149, 'r': 0.4555525807892092}, 'rouge-2': {'f': 0.12679050859440755, 'p': 0.09488362436389773, 'r': 0.21241284858146195}, 'rouge-l': {'f': 0.2268888663427004, 'p': 0.20887555329694585, 'r': 0.41535139227416007}}

weights -, novelty +

```
In [29]: model = SentenceTaggerRNN(vocab_size, use_novelty=True).to(device)
         train(use_class_weights=False, N_EPOCHS=N_EPOCHS, CLIP=CLIP)
```



Epoch: 05 | Time: 0m 9s

Train Loss: 0.232 | Train PPL: 1.261

Val. Loss: 0.235 | Val. PPL: 1.265

### top3

```
In [30]: model.load_state_dict(torch.load('best-val-model.pt'))
         inference_summarunner(model, test_iterator, top_k=3)
```

Count: 256

Ref: бразильская модель наджила триндаде , которая в 2019 году заявила , что ее изнасиловал нападающий « псж » неймар , выдвинула против него новое обвинение . согласно ее словам , футболист причастен к организации сети vip-проституции .

Нур: бразильский нападающий французского « псж » неймар снова может оказаться втянут в судебное разбирательство . иск против игрок

а намерена подать модель из бразилии наджила триндаде . по информации uol esporte , теперь девушка обвиняет форварда в организации сети vip-проституции .

BLEU: 0.44674925060378357

ROUGE: {'rouge-1': {'f': 0.3139099695755891, 'p': 0.2943140273165957, 'r': 0.35416244710992795}, 'rouge-2': {'f': 0.13833083065518084, 'p': 0.12774170969656698, 'r': 0.16134053532067022}, 'rouge-l': {'f': 0.2667387614491305, 'p': 0.2637614988963379, 'r': 0.3162068840205883}}

logits > 0

```
In [31]: inference_summarunner(model, test_iterator, top_k=None)
```

Count: 256

Ref: хакеры могут взломать смартфоны и « умные » колонки с помощью лазерной указки и пары других инструментов , заявили исследователи по кибербезопасности . сообщается , что некоторые модели гаджетов воспринимают свет от лазера как звуковую команду , а значит виртуальные голосовые помощники могут быть обмануты злоумышленниками , пожелавшими сделать покупку в интернете от вашего имени или открыть дверь гаража « умного » дома , чтобы угнать автомобиль .

Нур: группа исследователей из токийского университета электрокоммуникаций и университета мичигана обнаружила , что ряд гаджетов , поддерживающих голосовое управление , имеет в себе уязвимость – если навести на них лазерную указку , то они воспринимают свет как звуковую команду , сообщает business insider .

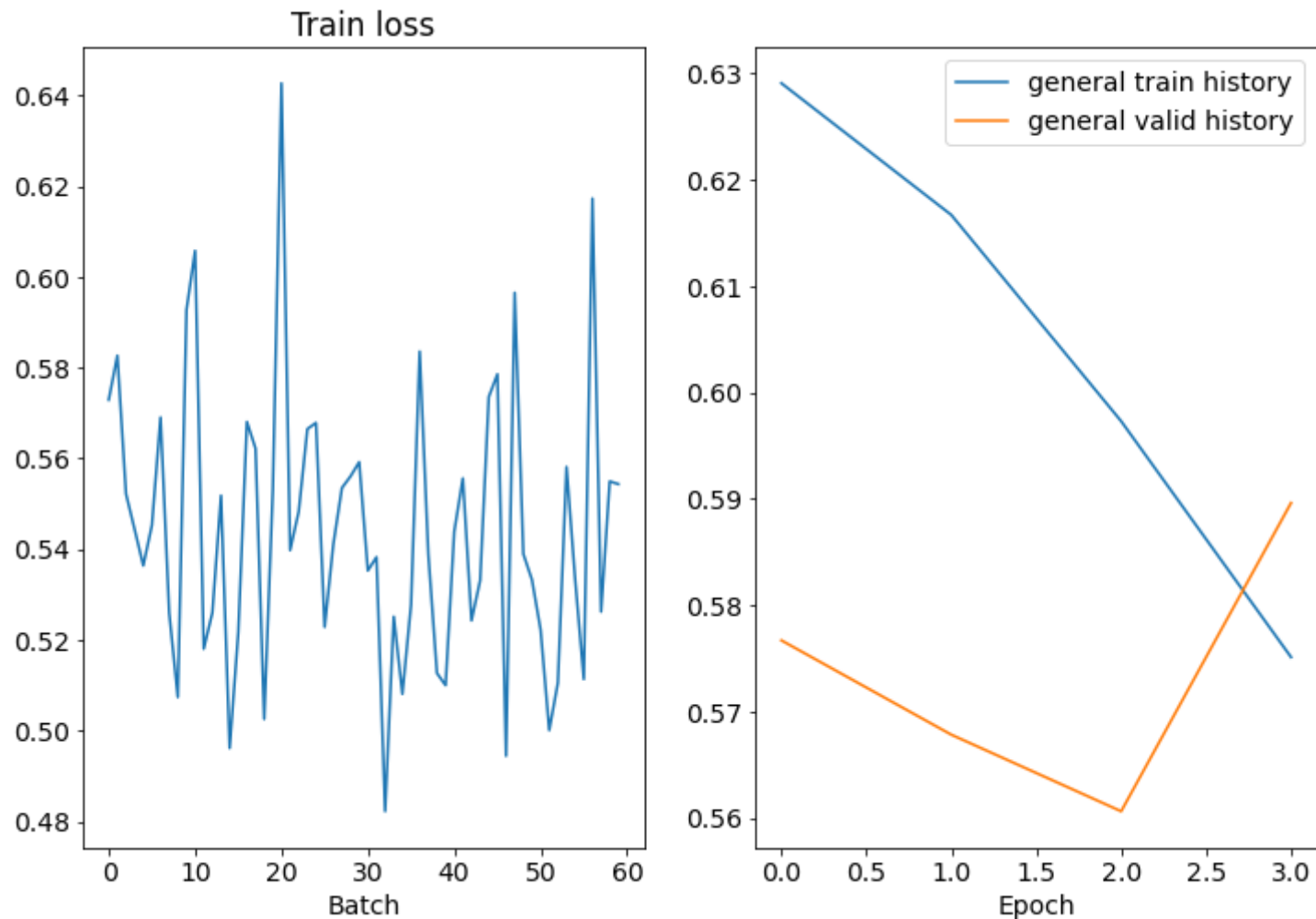
BLEU: 0.19968536742761978

ROUGE: {'rouge-1': {'f': 0.26750740225830005, 'p': 0.4255177372030778, 'r': 0.20709281205900218}, 'rouge-2': {'f': 0.1228254980375476, 'p': 0.20627572995628915, 'r': 0.09377851129323336}, 'rouge-l': {'f': 0.1946385291840424, 'p': 0.3714323303634145, 'r': 0.18004812342166793}}

## Without novelty

## weights +, novelty -

```
In [32]: model = SentenceTaggerRNN(vocab_size, use_novelty=False).to(device)
          train(use_class_weights=True, N_EPOCHS=N_EPOCHS, CLIP=CLIP)
```



Epoch: 05 | Time: 0m 8s

Train Loss: 0.545 | Train PPL: 1.724

Val. Loss: 0.582 | Val. PPL: 1.789

### top3

```
In [33]: model.load_state_dict(torch.load('best-val-model.pt'))
         inference_summarunner(model, test_iterator, top_k=3)
```

Count: 256

Ref: владельцы китайских ресторанов и магазинчиков в сша не знают , как будут платить аренду в следующем месяце – продажи катастрофически падают . американцы начали бояться жареной лапши и дим-самов из-за распространения коронавируса нового типа – они больше не ходят в заведения китайской кухни , и властям городов приходится уговаривать жителей не поддаваться истерии .

Нур: китайские рестораны становятся жертвами коронавируса нового типа : в сша в заведениях , где подают блюда китайской кухни , обыватели начали видеть источник заразы . американцы не просто начали обходить стороной рестораны , где подают дим-самы , жареную лапшу и свинину в кисло-сладком соусе , но и призывают других это не делать , а владельцы некогда популярных ресторанов и закусочных подсчитывают убытки и терпят расистские нападки . в американском штате аризона из-за коронавируса covid-19 под угрозой оказался гастрономический фестиваль в городе меса , где недавно был создан отдельный кластер с ресторанами азиатской кухни .

BLEU: 0.4468518049027453

ROUGE: {'rouge-1': {'f': 0.3190150135053712, 'p': 0.29857671484432463, 'r': 0.36075073392896195}, 'rouge-2': {'f': 0.1411717652336616, 'p': 0.13012767053720578, 'r': 0.16496109505818607}, 'rouge-l': {'f': 0.27137751189281917, 'p': 0.2677933797965847, 'r': 0.32276683072582857}}

## logits > 0

```
In [34]: inference_summarunner(model, test_iterator, top_k=None)
```

Count: 256

Ref: на кубани в городе кропоткине более 50 школьников госпитализированы с признаками острой кишечной инфекции – все дети от семи до 15 лет находятся в состоянии средней тяжести . следственные органы возбудили уголовное дело по статье об оказании недоброкачественных услуг – родители считают , что причиной отравления стала селедка , которую накануне давали в школьной столовой .

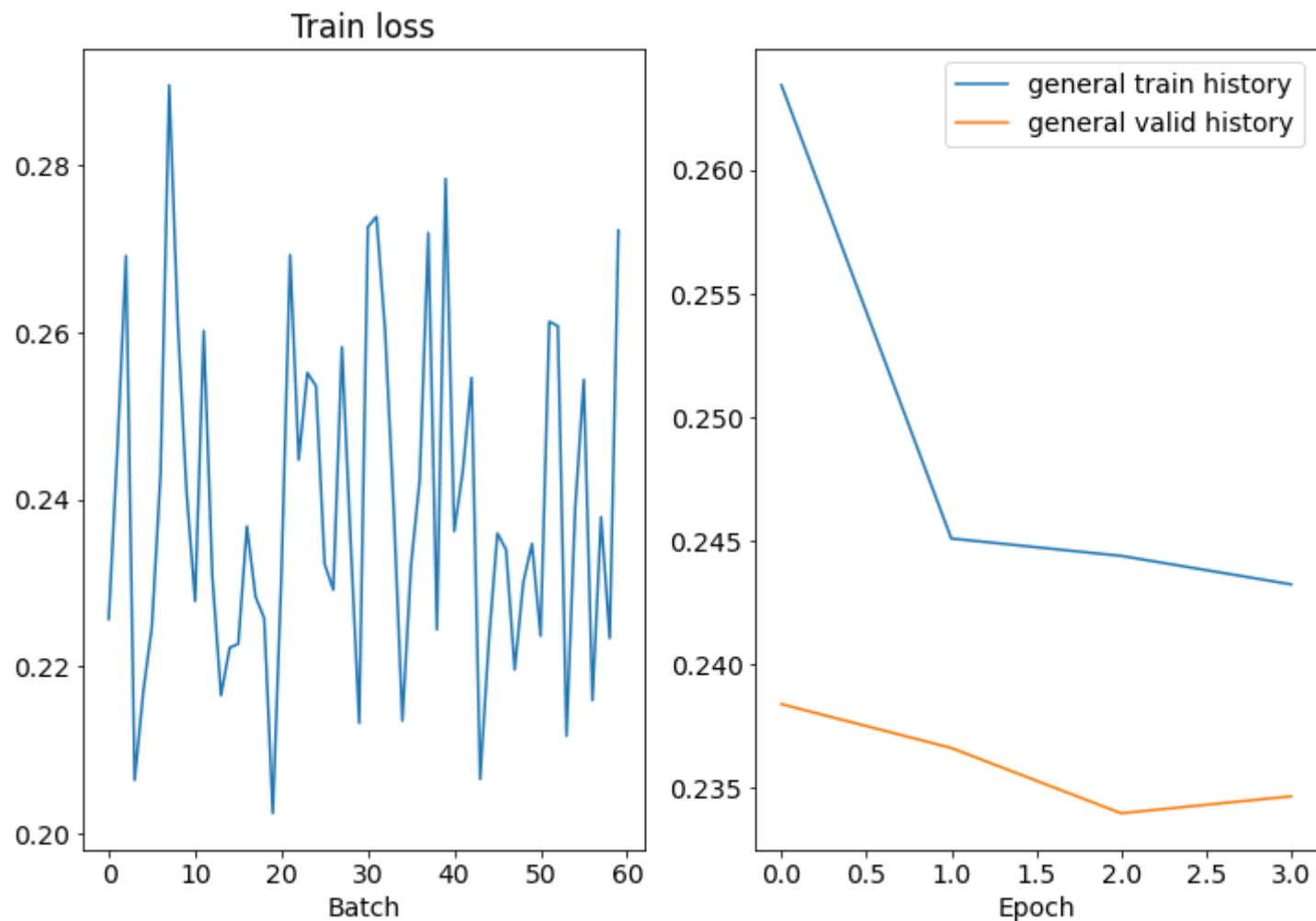
Нур: в городе кропоткине краснодарского края более 50 школьников отравились в школьных столовых – дети от семи до 15 лет сейчас находятся в детском инфекционном отделении горбольницы . об этом сообщает издание « кп » - кубань » . по данным сми , об инциденте стало известно 9 ноября 2019 года . в этот день 28 школьников с подозрением на отравление доставили в больницу – все они были в состоянии средней тяжести . однако 10 ноября госпитализировали уже 51 человек . 10 человек доставили в кавказскую центральную районную больницу , 41 – в кропоткинскую городскую . « контактных пациентов осматривают на дому 6 медицинских бригад . « управлением Роспотребнадзора по краснодарскому краю проводится санитарно-эпидемиологическое расследование и комплекс противоэпидемических мероприятий .

BLEU: 0.26329591404597413

ROUGE: {'rouge-1': {'f': 0.27916498393617, 'p': 0.20348127575297614, 'r': 0.4783487346752719}, 'rouge-2': {'f': 0.11652257581862352, 'p': 0.08203224167719596, 'r': 0.22198005208366048}, 'rouge-l': {'f': 0.20144819033327263, 'p': 0.18497152372683223, 'r': 0.4354726201388017}}

## weights -, novelty -

```
In [35]: model = SentenceTaggerRNN(vocab_size, use_novelty=False).to(device)
         train(use_class_weights=False, N_EPOCHS=N_EPOCHS, CLIP=CLIP)
```



Epoch: 05 | Time: 0m 8s

Train Loss: 0.238 | Train PPL: 1.269

Val. Loss: 0.233 | Val. PPL: 1.263

### top3

```
In [36]: model.load_state_dict(torch.load('best-val-model.pt'))
         inference_summarunner(model, test_iterator, top_k=3)
```

Count: 256

Ref: из какого раствора можно надуть гигантские мыльные пузыри , методом проб и ошибок выяснили американские ученые . предложенным рецептом они поделились со взрослыми и детьми .

Нур: все дети и взрослые обожают надувать мыльные пузыри . однако немногие задумывались над тем , что надо сделать , чтобы надуть

по истине гигантские пузыри и какого размера они вообще могут быть . такая проблема заняла ученых из университета эмори ( сша ) по д руководством джастина бертона , который занимается проблемами динамики жидкостей в своей лаборатории .

BLEU: 0.43275192972528337

ROUGE: {'rouge-1': {'f': 0.305247650354594, 'p': 0.28208589356655794, 'r': 0.3503887279355527}, 'rouge-2': {'f': 0.13369778758215548, 'p': 0.12128549164708496, 'r': 0.15939716640943574}, 'rouge-l': {'f': 0.2586058109346039, 'p': 0.25274485208891756, 'r': 0.31358682088740497}}

logits > 0

In [37]: inference\_summarunner(model, test\_iterator, top\_k=None)

Count: 256

Ref: французская актриса валентина монньер выступила с обвинениями в адрес режиссера романа поланского . по ее словам , в 1975 году он жестоко изнасиловал ее на курорте в швейцарии , когда девушке только исполнилось 18 лет . монньер подчеркнула , что решила открыто рассказать о произошедшем из-за попыток режиссера оправдаться за прежние обвинения в своем новом фильме .

Нур: с новыми обвинениями в сексуальном насилии со стороны режиссера романа поланского выступила французская актриса валентина монньер , пишет le parisien .

BLEU: 0.08639987994055964

Error by ROUGE score calculation: Collections must contain at least 1 sentence.

## Вывод:

- величина BLEU 0.45 (с округлением до 2-го знака) была достигнута
- максимальный BLEU (~0.45) был при параметрах - с весами без novelty, top3
- аналогичные BLEU (с точностью до 3 знака) получены также при параметрах - все top3, без весов с novelty, с весами и с novelty
- веса, получается, сильно не повлияли
- BLEU без весов и без novelty был чуть ниже, ~0.43
- novelty повлияло только на вариант без весов, при варианте с весами novelty сильного влияния не оказало
- при расчете скоров по только положительным логитам значения скоров сильно падали. Иногда вариантов с положительными логитами вообще не получалось. Возможно, эти два обстоятельства связаны между собой