Abstractive Summarization

Wikipedia

HARUN EREN | RALF KINKEL | DIEGO MIGUEL

Introduction

Abstractive summarization = summaries with novel phrases and words.

Best artificial abstractive summaries done with neural network,
but with a huge gap to human performance!

Choosing a loss function

Neural networks need loss function to train.

Next word prediction is simple but is overly restrictive and suffers from "exposure bias".

Human experts would give best feedback on how good a summary is, but that's not doable.

=> Using another model for evaluating summaries and creating loss in GAN approach.

Last hurdle: Non-differentiablity

Tackling the Non-Differentiability Problem

Introducing our novel approach

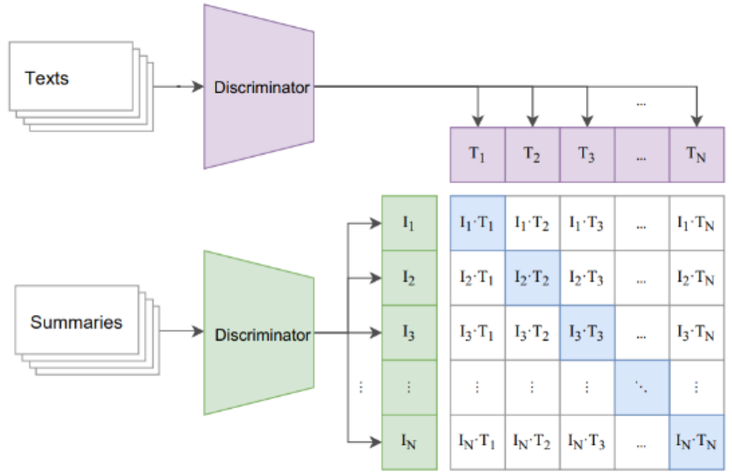
Tackling Non-Differentiability

Sampling multiple tokens, evaluating all resulting summaries and creating cross entropy loss from evaluations of discriminator and next token predictions of summarizer.

Overview approach

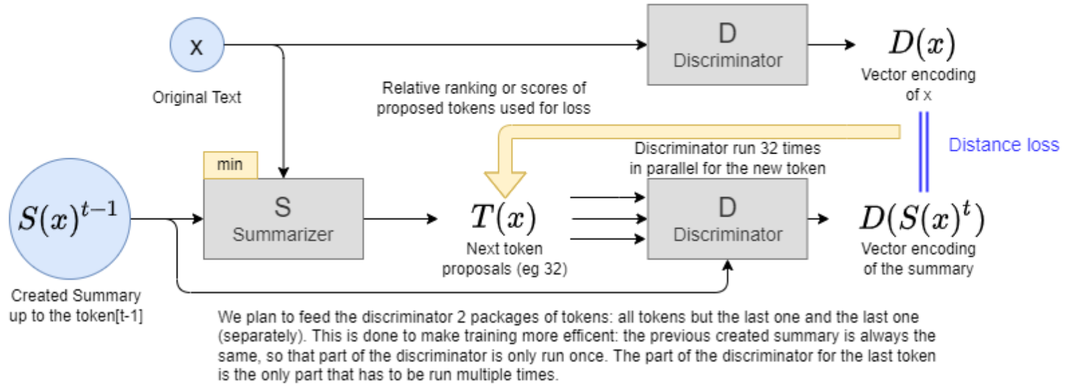
Step 0: Pre-training of discriminator

(1) Contrastive pre-training



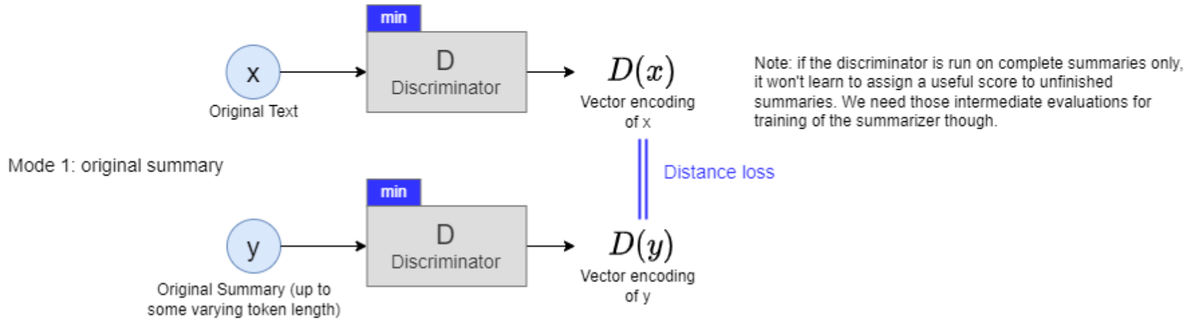
Before starting the iterative GAN-training, you can first spend a short time improving the discriminator with an approach taken from the OpenAI CLIP paper:
You can optimize the discriminator by encouraging similarity in the output vector for matching texts and summaries (the diagonal) and penalizing similarity in outputs for non-matching texts and summaries (everything else).
The output vector of the discriminator is normalized to 1 so that the value of I1*T1 is between -1 and 1.

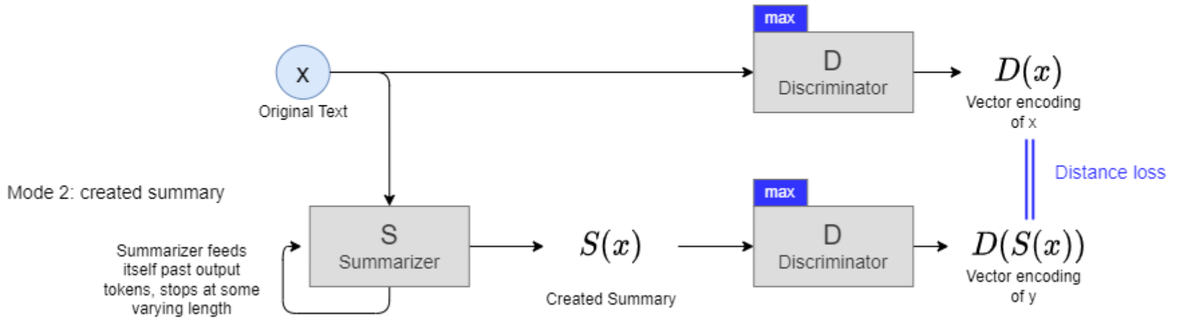
Step 1: Training the summary generator



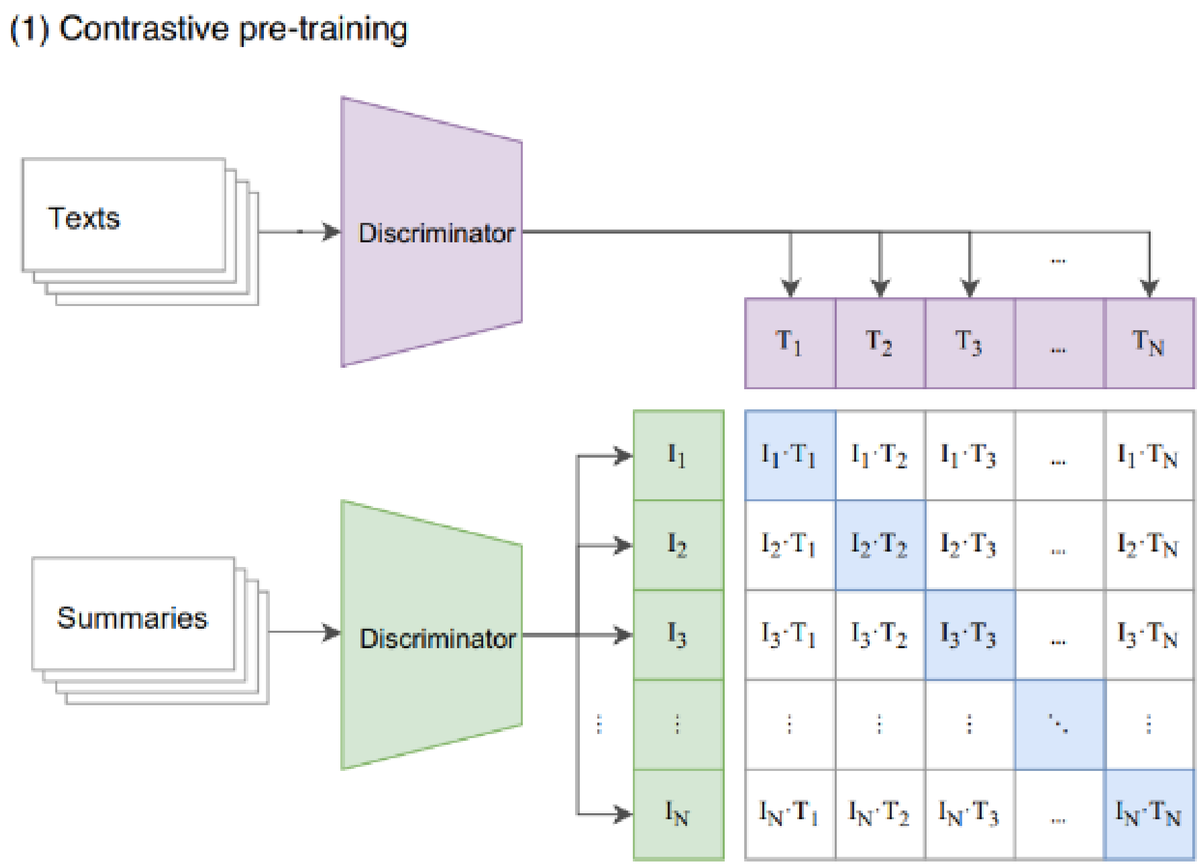
We plan to feed the discriminator 2 packages of tokens: all tokens but the last one and the last one (separately). This is done to make training more efficent: the previous created summary is always the same, so that part of the discriminator is only run once. The part of the discriminator for the last token is the only part that has to be run multiple times.

Step 2: Training the discriminator

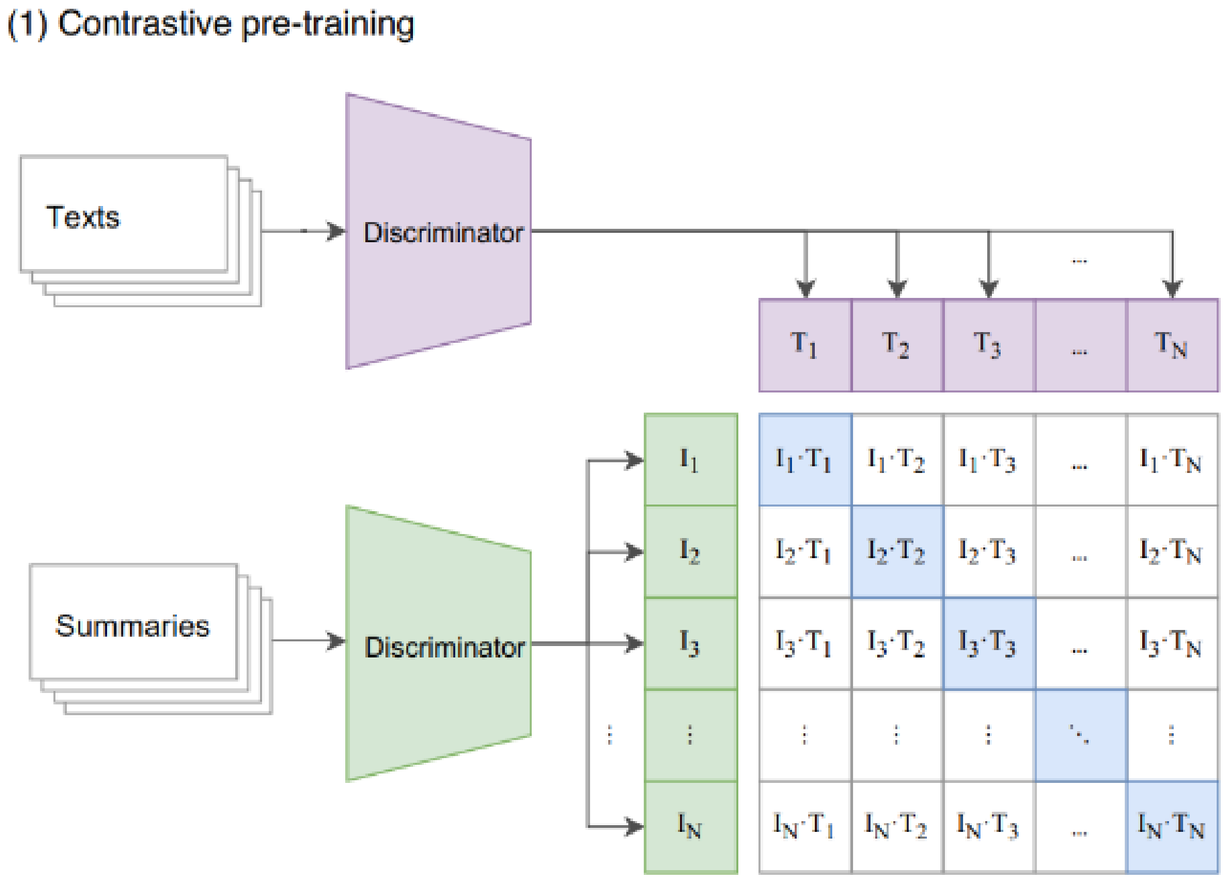




Step 0: Pre-training of discriminator

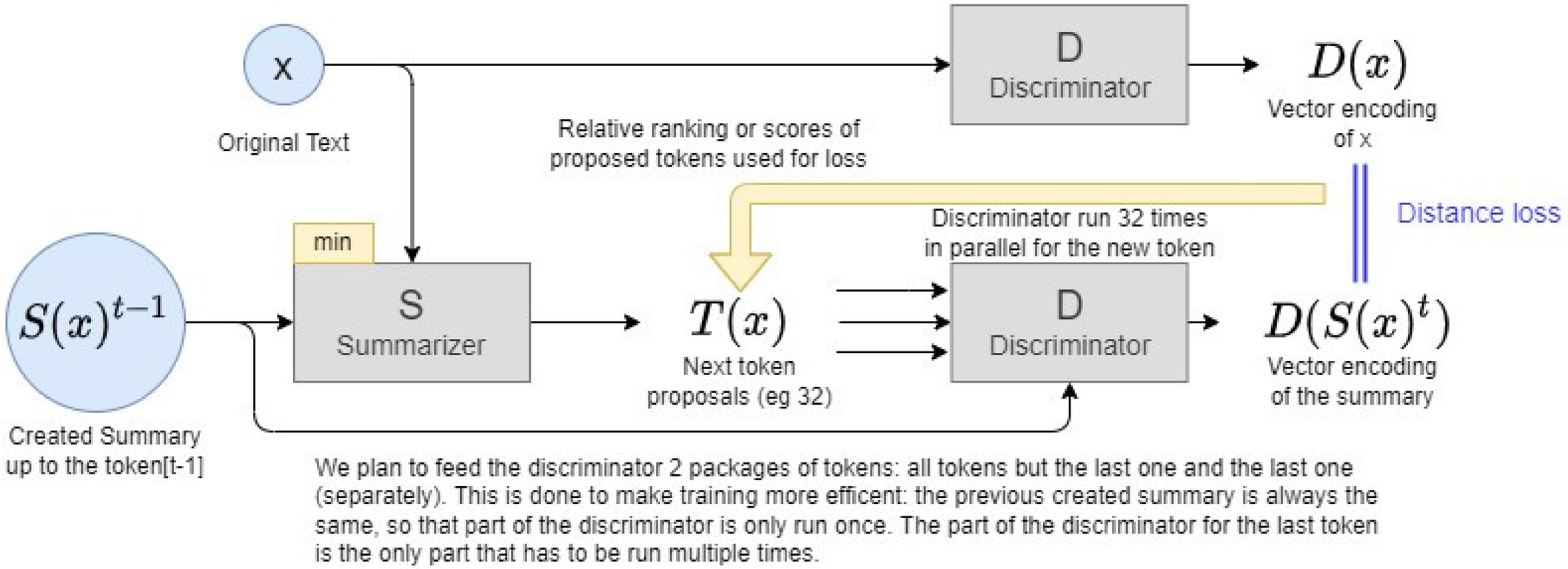


Step 0: Pre-training of discriminator



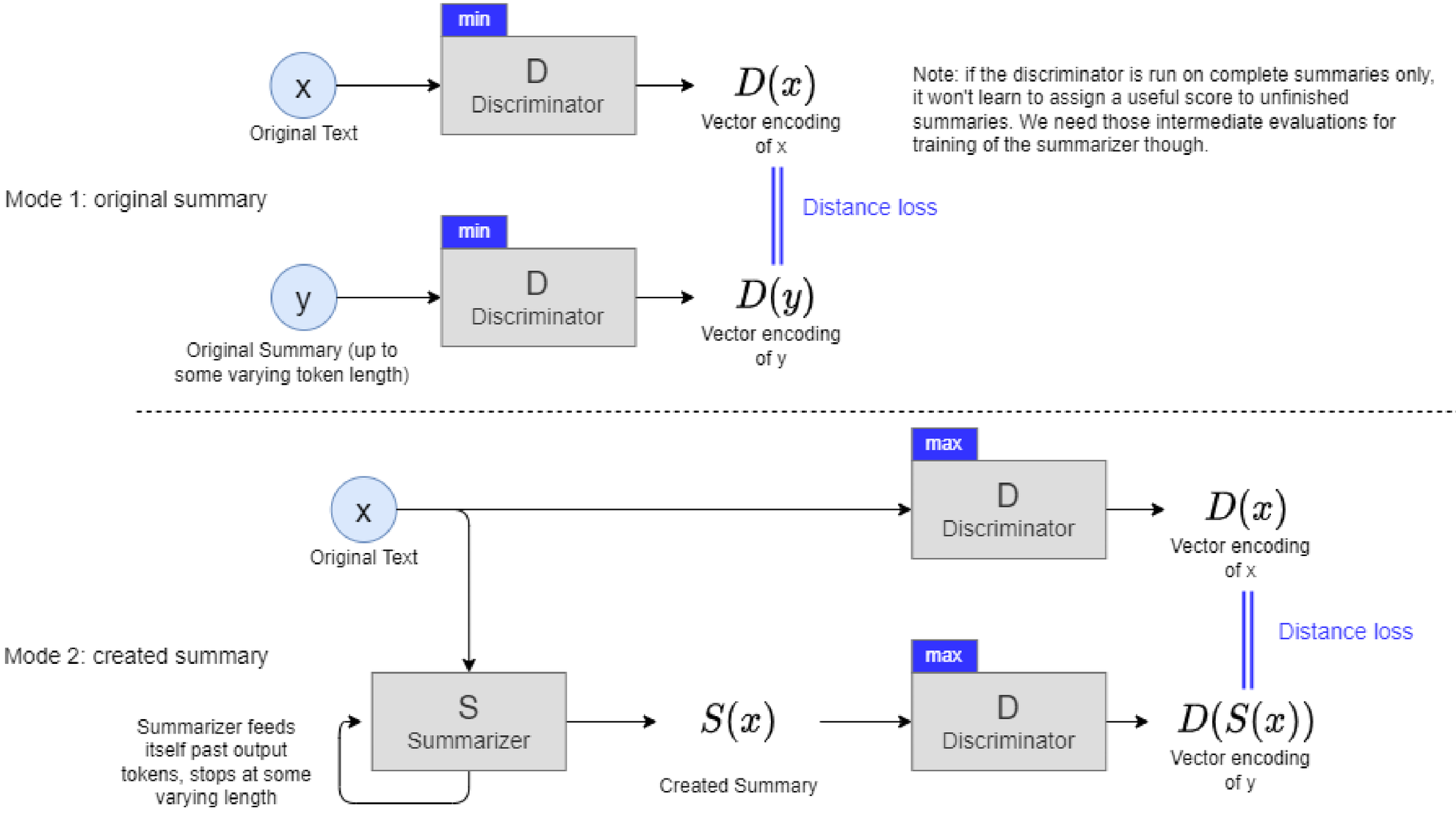

```
Epoch = 0
[[0.25 -0.71 -0.67]
 [0.12 -0.54 -0.23]
 [-0.43 0.27 0.25]]

Epoch = 10
[[0.90 -0.54 -0.78]
 [-0.55 0.97 -0.11]
 [-0.69 -0.12 0.88]]

Epoch = 20
[[0.99 -0.28 -0.52]
 [-0.36 0.98 -0.58]
 [-0.67 -0.56 0.98]]

Epoch = 30
[[1.00 -0.45 -0.45]
 [-0.50 1.00 -0.57]
 [-0.46 -0.57 1.00]]

Epoch = 40
[[1.00 -0.57 -0.48]
 [-0.53 0.99 -0.49]
 [-0.41 -0.51 0.99]]
```

Step 1: Training of the summary generator



We plan to feed the discriminator 2 packages of tokens: all tokens but the last one and the last one (separately). This is done to make training more efficent: the previous created summary is always the same, so that part of the discriminator is only run once. The part of the discriminator for the last token is the only part that has to be run multiple times.

Step 2: Training of the discriminator

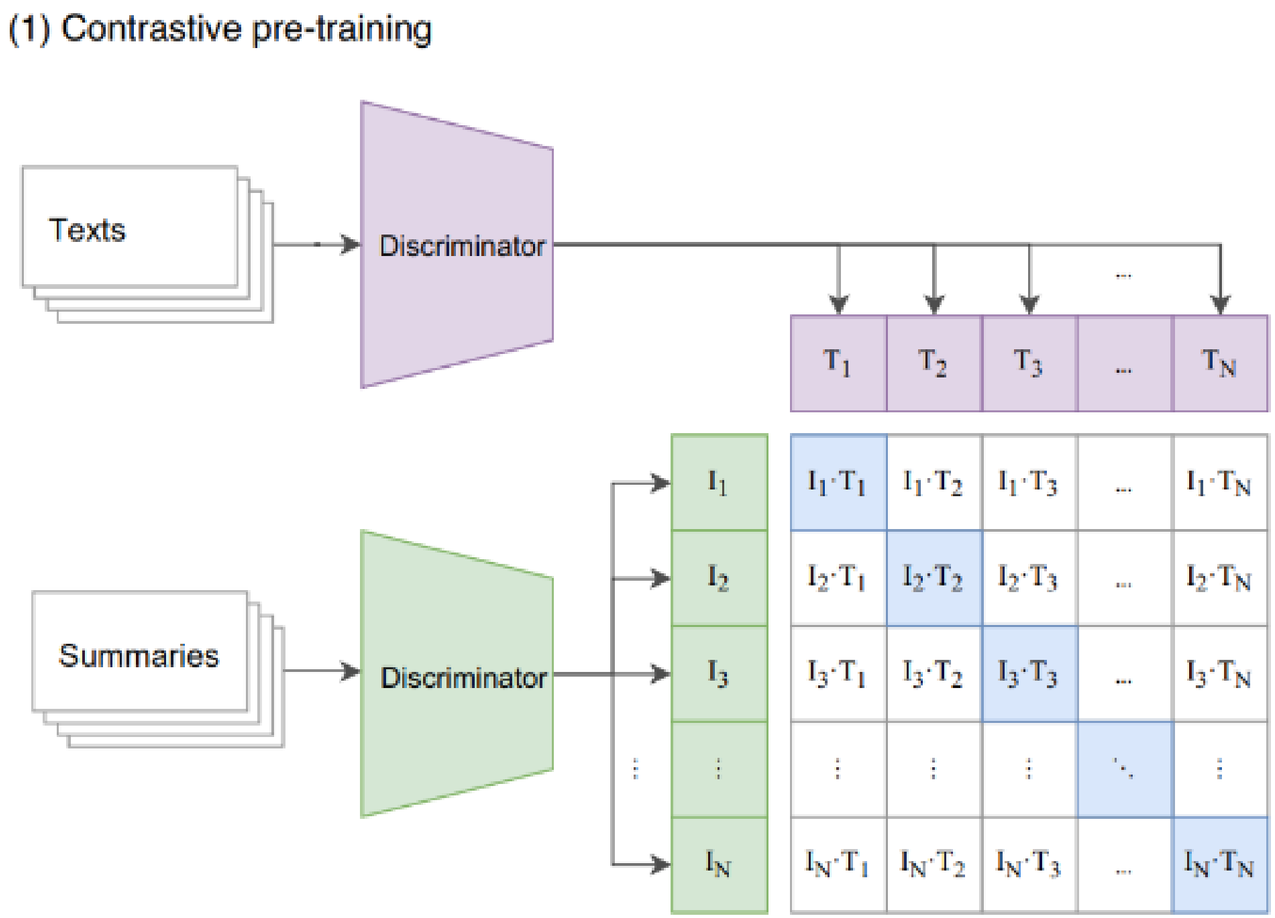


Our progress so far...

1

Pretraining of the Discriminator

Pre-training of discriminator



First try: Pretrain ourselves => switch to using pretrained model and refining here instead.

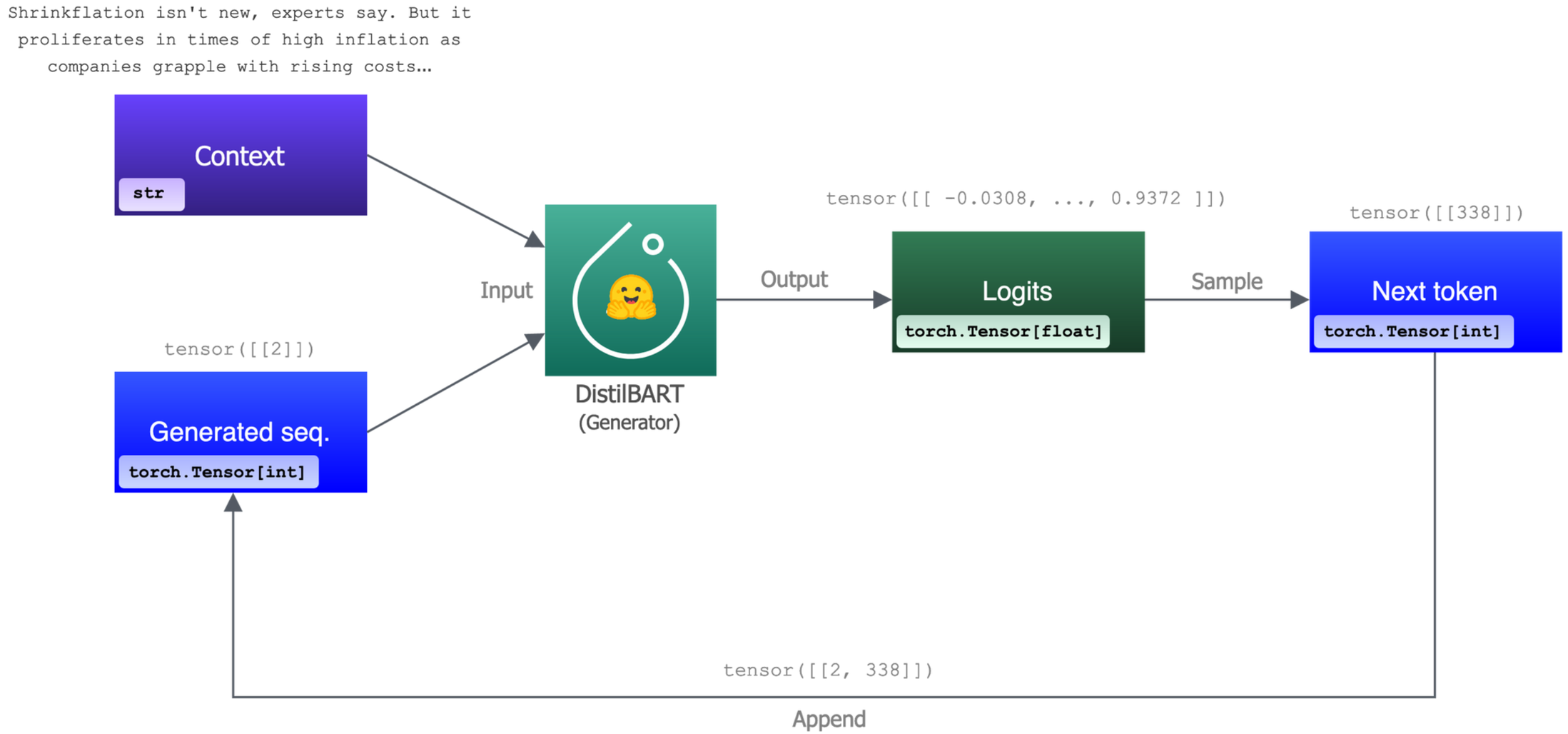
MPNet variant pretrained for sentence similarity used.

Refining step improved loss and we got first promising results!

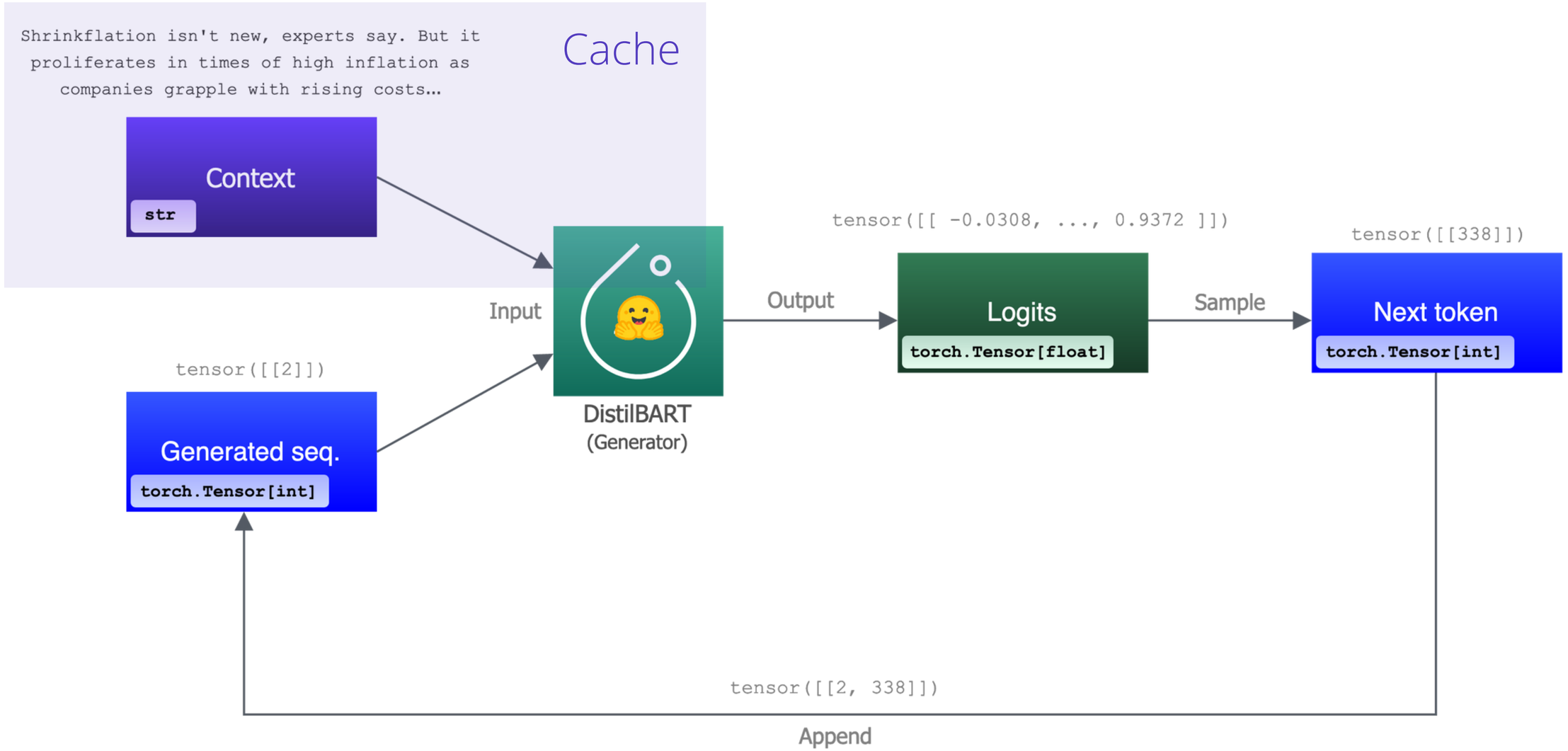
2

Controlling Summary Generation

Controlling Summary Generation (I)



Controlling Summary Generation (II)



Controlling Summary Generation – Cache

Our pipeline

```
['</s> The US has announced that it is to be a "big box" for the first time in 12 years.</s>']

Generation time: 4.083 seconds
```

Controlling Summary Generation – Cache

Our pipeline

```
['</s> The US has announced that it is to be a "big box" for the first time in 12 years.</s>']

Generation time: 4.083 seconds
```

 pipeline

```
['</s> The US has announced that it is to be a "big box" for the first time in 12 years.</s>']

Generation time: 0.501 seconds
```

Controlling Summary Generation – Cache

We will simply modify existent HuggingFace code.

Controlling Summary Generation – Cache

We will simply modify existent HuggingFace code.

3

Datasets

Progress: Datasets (I)

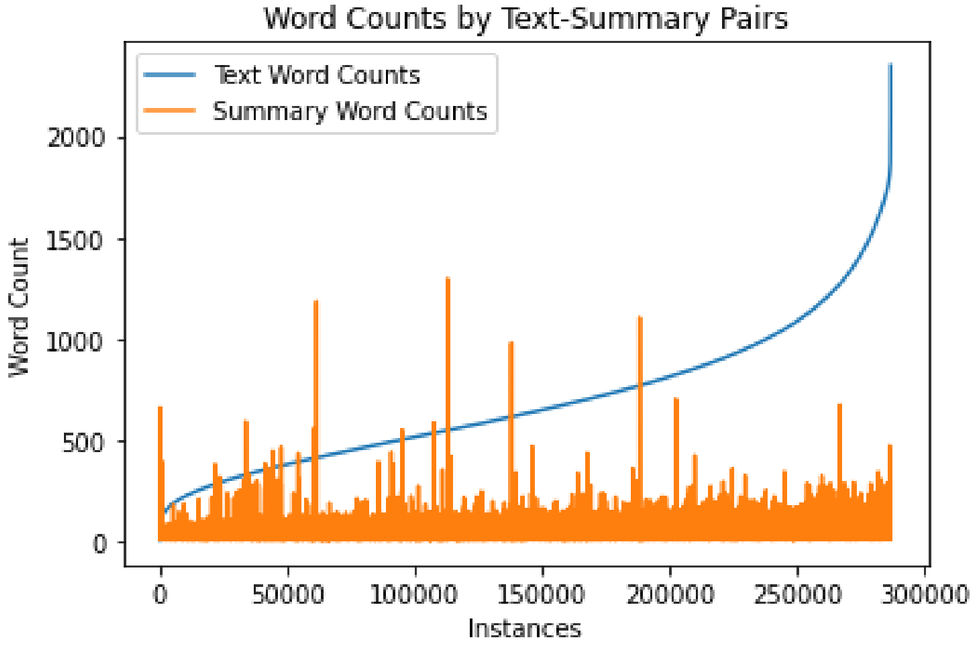
CNN & DM Dataset

XSum Dataset

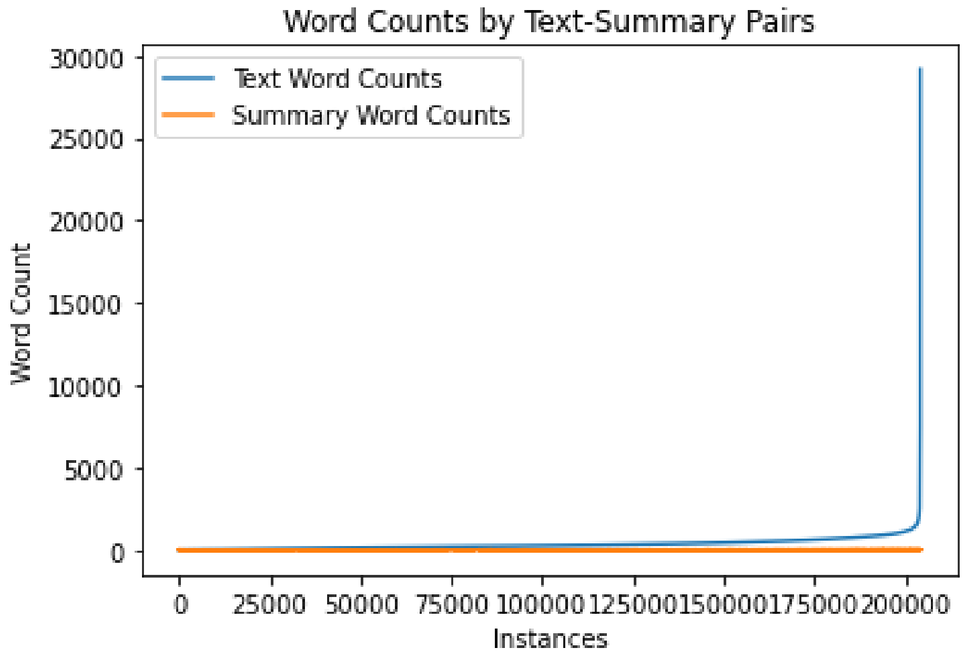
Hugging Face

Progress: Datasets (II)

CNN & DM Dataset

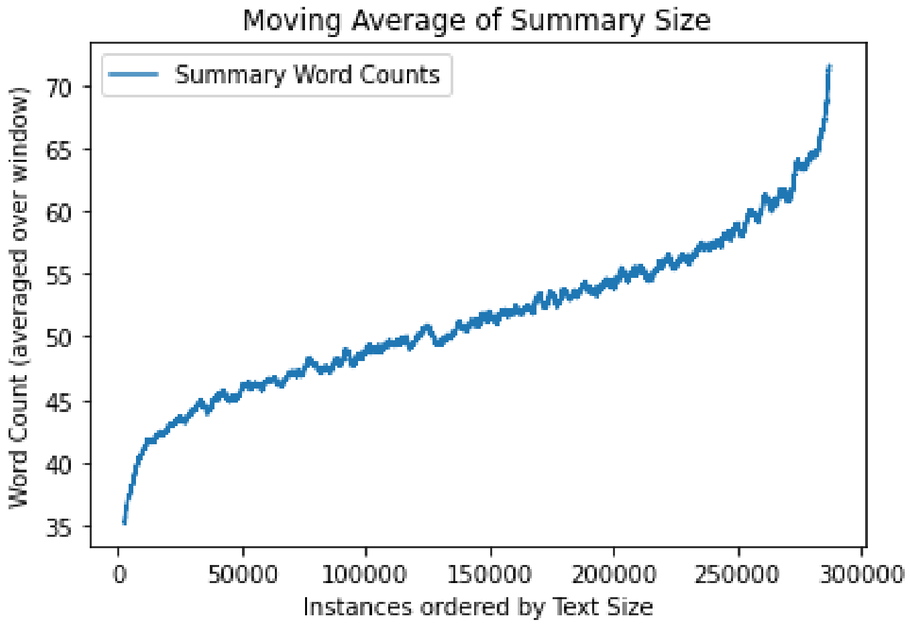


XSum Dataset

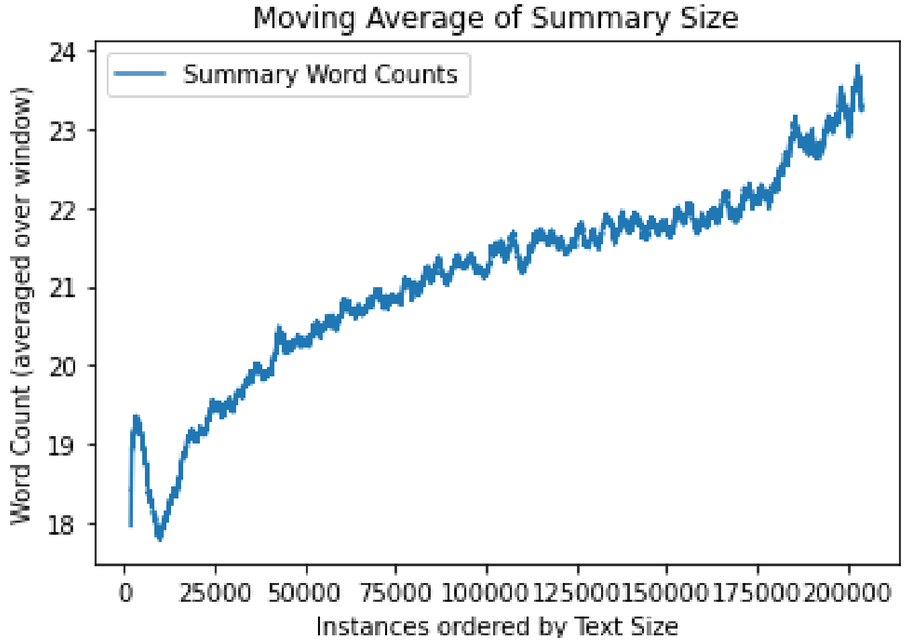


Progress: Datasets (III)

CNN & DM Dataset



XSum Dataset

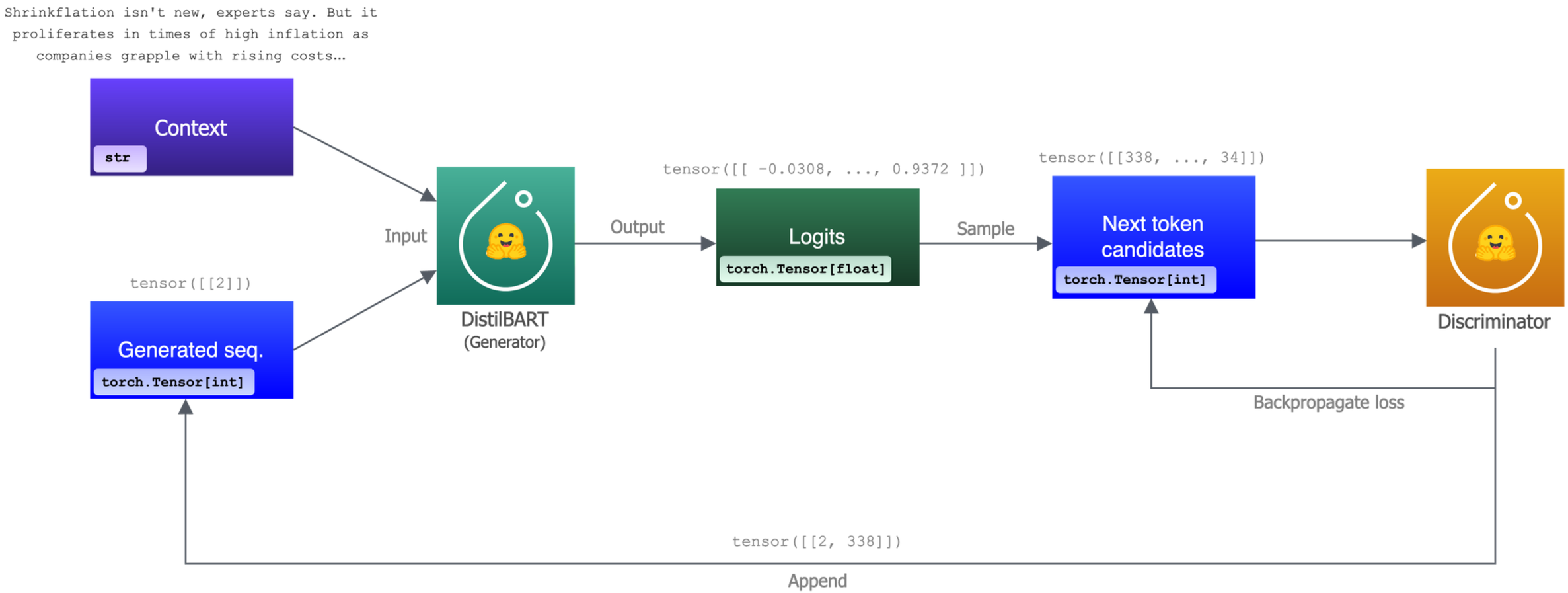


Next steps...

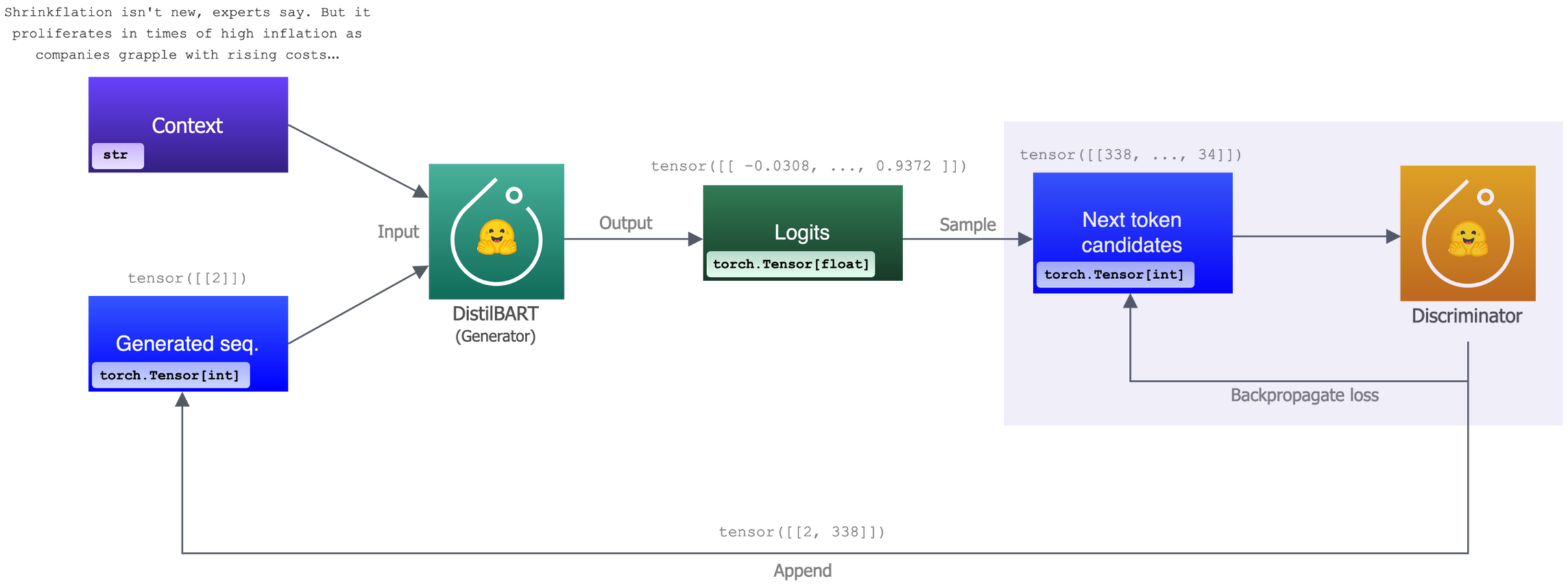
1

Training / Fine-tuning the Generator

Training / Fine-tuning the Generator



Training / Fine-tuning the Generator



2

Training / Fine-tuning of Discriminator

3

Putting Everything Together

4

Evaluation

5

Wikipedia Dataset

6

Scaling Up the Architecture

6

Scaling Up the Architecture

6

Scaling Up the Architecture

Abstractive Summarization

Wikipedia

Thank you for your attention.

HARUN EREN | RALF KINKEL | DIEGO MIGUEL

Appendix: Discriminator first results

```
example_text = ["""SAN FRANCISCO, California (CNN) -- A magnitude 4.2 earthquake shook the San Francisco area Friday at 4:42 a.m. PT (7:42 a.m. ET),
the U.S. Geological Survey reported. The quake left about 2,000 customers without power, said David Eisenhower, a spokesman for Pacific Gas and Light.
Under the USGS classification, a magnitude 4.2 earthquake is considered "light," which it says usually causes minimal damage. "We had quite a spike in calls,
mostly calls of inquiry, none of any injury, none of any damage that was reported," said Capt. Al Casciato of the San Francisco police.
"It was fairly mild." Watch police describe concerned calls immediately after the quake » . The quake was centered about two miles east-northeast of Oakland,
at a depth of 3.6 miles, the USGS said. Oakland is just east of San Francisco, across San Francisco Bay. An Oakland police dispatcher told CNN the quake set off alarms at people's homes.
The shaking lasted about 50 seconds, said CNN meteorologist Chad Myers. According to the USGS, magnitude 4.2 quakes are felt indoors and may break dishes and windows and overturn unstable objects.
Pendulum clocks may stop. E-mail to a friend."""]

example_summaries = [["Big earthquake hits San Francisco, thousands without power"],
                     ["Marvel reveals new superhero"],
                     ["Al Carciato of the San Francisco police says 'it's fairly mild"],
                     ["Police car chases 4.2 earthquake in San Francisco"],
                     ["4:42 a.m. PT (7:42 a.m. ET) David Eisenhower Capt. Al Casciato two miles east-northeast USGS CNN"]]

for summary in example_summaries:
  print(summary_fit(example_text, summary, post_step_0=False))
  print(summary_fit(example_text, summary, post_step_0=True))
  print()
```

```
tensor([0.7444])
tensor([0.9921])

tensor([-0.0058])
tensor([0.0282])

tensor([0.3344])
tensor([-0.0737])

tensor([0.5789])
tensor([0.9562])

tensor([0.1785])
tensor([-0.0460])
```