트랜스포머를 활용한 자연어 처리 5, 6장

루이 턴스톨, 레안트로 폰 베라, 토마스 울프 지음

박해선 옮김

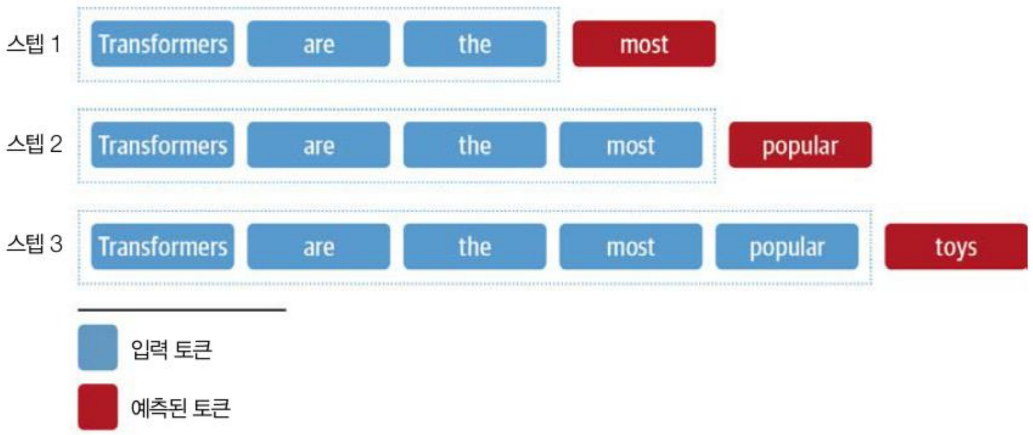
발제자 : 정현우 (junghw3333@gmail.com)

랩 : HUMANE Lab

2024-01-16

텍스트 생성: inference

P(next tokens | context)



스텝 1 Transformers are the most

스텝 2 Transformers are the most popular

스텝 3 Transformers are the most popular toys

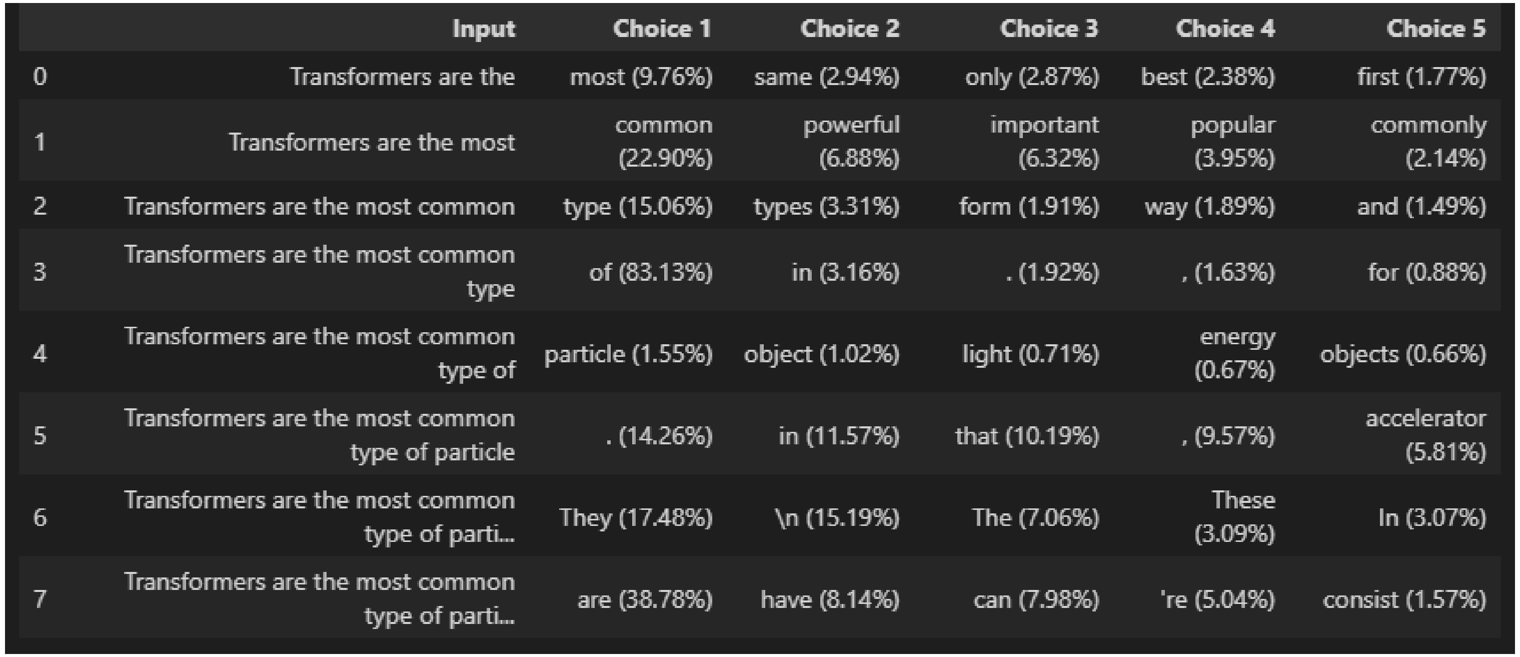
🟦 입력 토큰

🟥 예측된 토큰

=> ∏ P(next tokens | previous token)

그리디(Greedy) 서치 디코딩

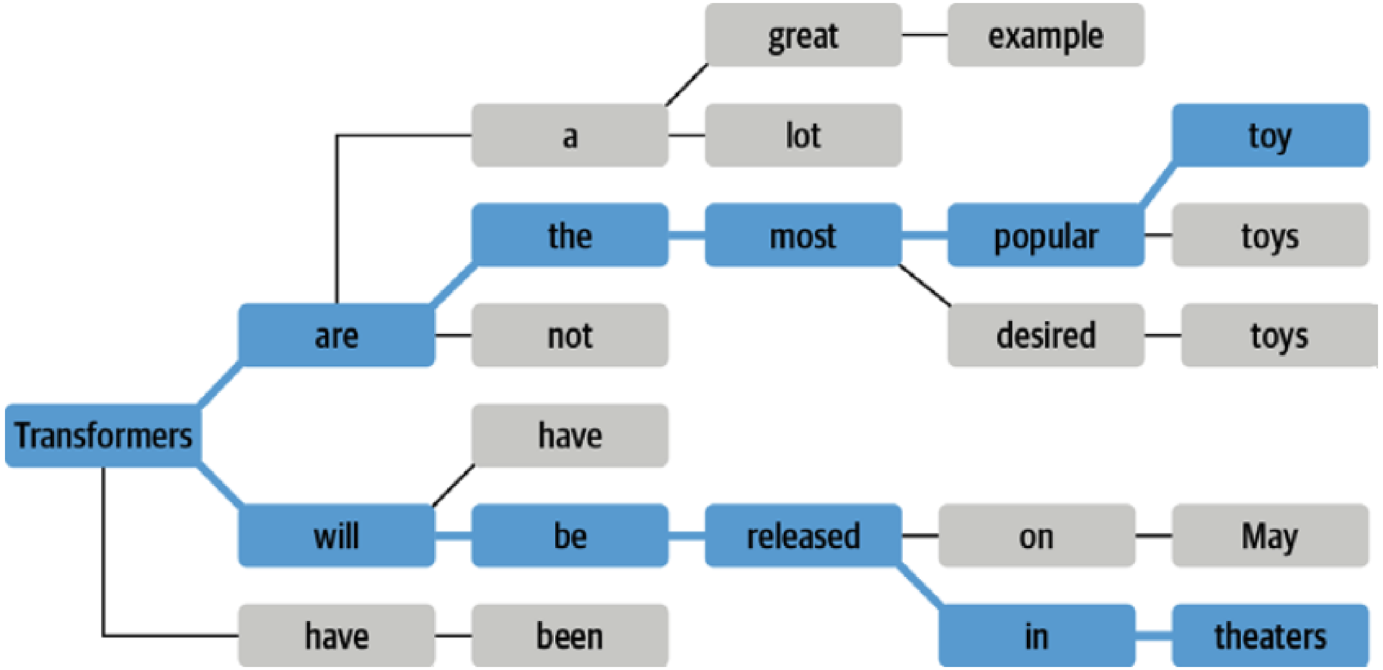
P(next tokens | context)의 모든 가능한 경우의 수는 너무 많음.
때문에 연속적으로 Choice 1을 선택해서 문장을 완성함.



	Input	Choice 1	Choice 2	Choice 3	Choice 4	Choice 5
0	Transformers are the	most (9.76%)	same (2.94%)	only (2.87%)	best (2.38%)	first (1.77%)
1	Transformers are the most	common (22.90%)	powerful (6.88%)	important (6.32%)	popular (3.95%)	commonly (2.14%)
2	Transformers are the most common	type (15.06%)	types (3.31%)	form (1.91%)	way (1.89%)	and (1.49%)
3	Transformers are the most common type	of (83.13%)	in (3.16%)	. (1.92%)	, (1.63%)	for (0.88%)
4	Transformers are the most common type of	particle (1.55%)	object (1.02%)	light (0.71%)	energy (0.67%)	objects (0.66%)
5	Transformers are the most common type of particle	. (14.26%)	in (11.57%)	that (10.19%)	, (9.57%)	accelerator (5.81%)
6	Transformers are the most common type of parti...	They (17.48%)	\n (15.19%)	The (7.06%)	These (3.09%)	In (3.07%)
7	Transformers are the most common type of parti...	are (38.78%)	have (8.14%)	can (7.98%)	're (5.04%)	consist (1.57%)

빔(Beam) 서치 디코딩

특정 시점에서 선택하는 것이 아닌, EOS까지의 확률로 선택함.
때문에 그리디 방식보다 문장에 대한 확률이 더 높음.

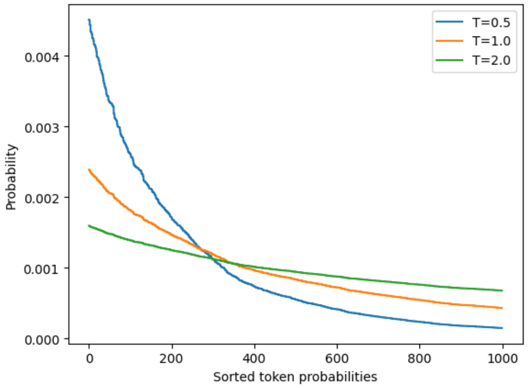


샘플링 방법

- **온도 상수 이용**
  - **수식**

$$P(y_t = w_i \mid y_{<t},\ x) = \frac{\exp(z_{t,i}/T)}{\sum_{j=1}^{|V|}\ \exp(z_{t,j}/T)}$$

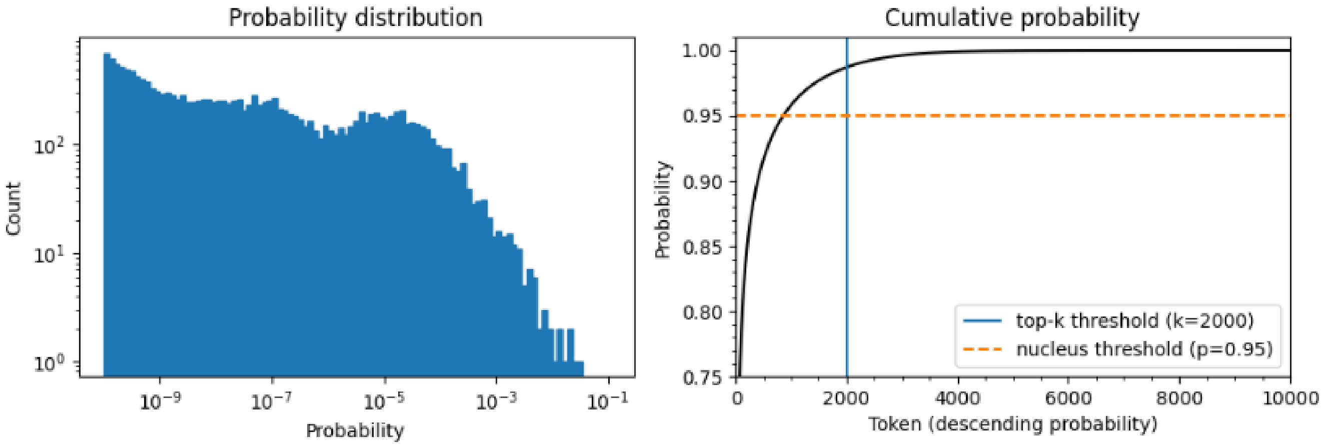
  - **T값이 높을수록 다양한 표현이 나옴, T값이 낮을수록 자주 쓰는 표현이 나옴.**



  - **둘은 Trade-off 관계로 상황에 따라 조정이 필요함.**

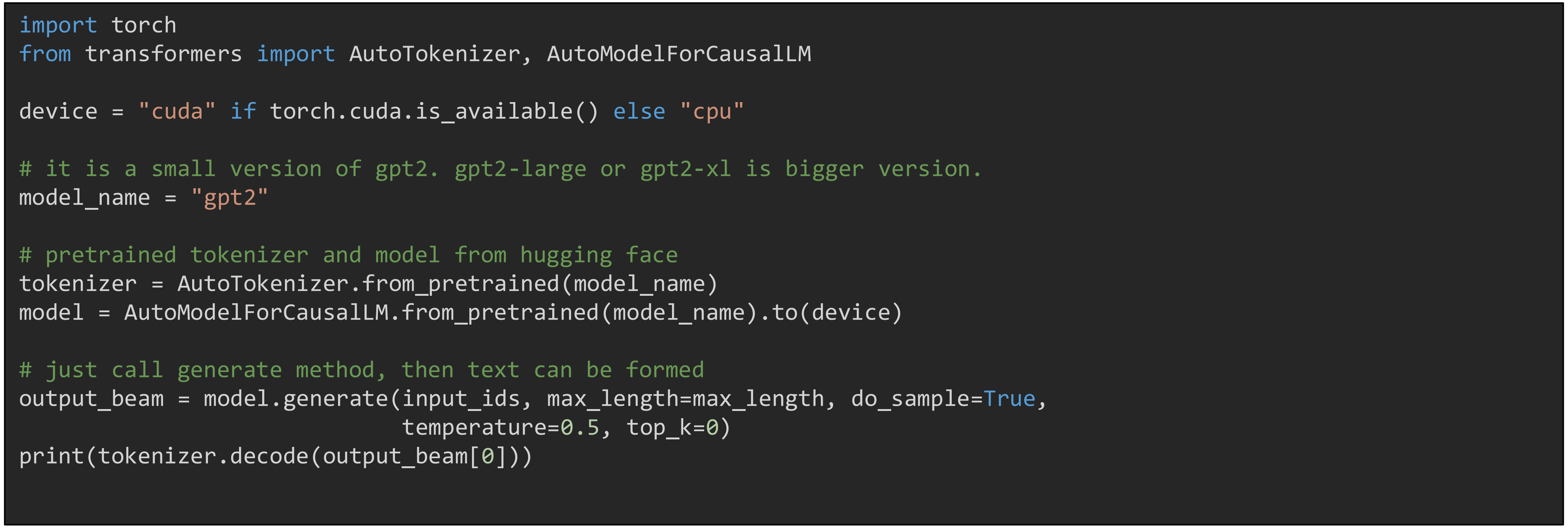
샘플링 방법

- **Top-k 이용**
  - **상위 k개의 단어로 대부분의 표현을 얻을 수 있다는 생각이 깔려 있음.**

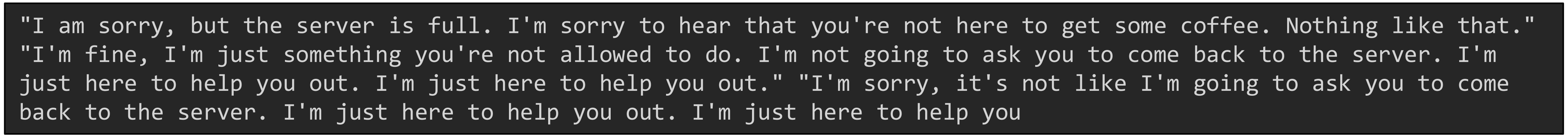


  - **때문에 출현 확률 값이 상위 k개의 단어만 선택하여 문장을 구성할 수 있음.**

How to use?

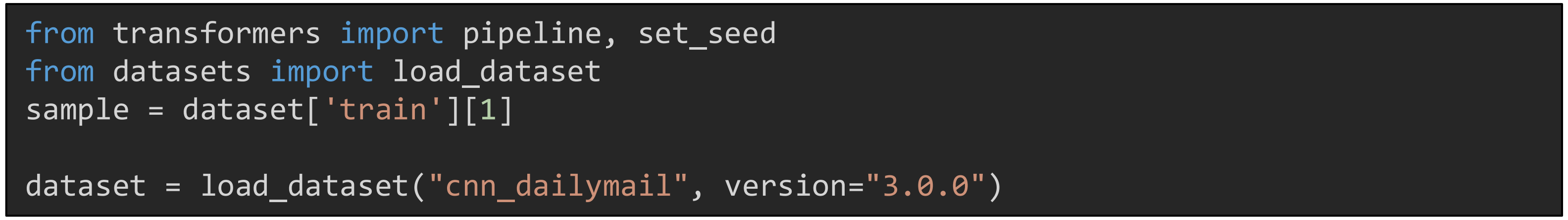


```python
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM

device = "cuda" if torch.cuda.is_available() else "cpu"

# it is a small version of gpt2. gpt2-large or gpt2-xl is bigger version.
model_name = "gpt2"

# pretrained tokenizer and model from hugging face
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name).to(device)

# just call generate method, then text can be formed
output_beam = model.generate(input_ids, max_length=max_length, do_sample=True,
                             temperature=0.5, top_k=0)
print(tokenizer.decode(output_beam[0]))
```

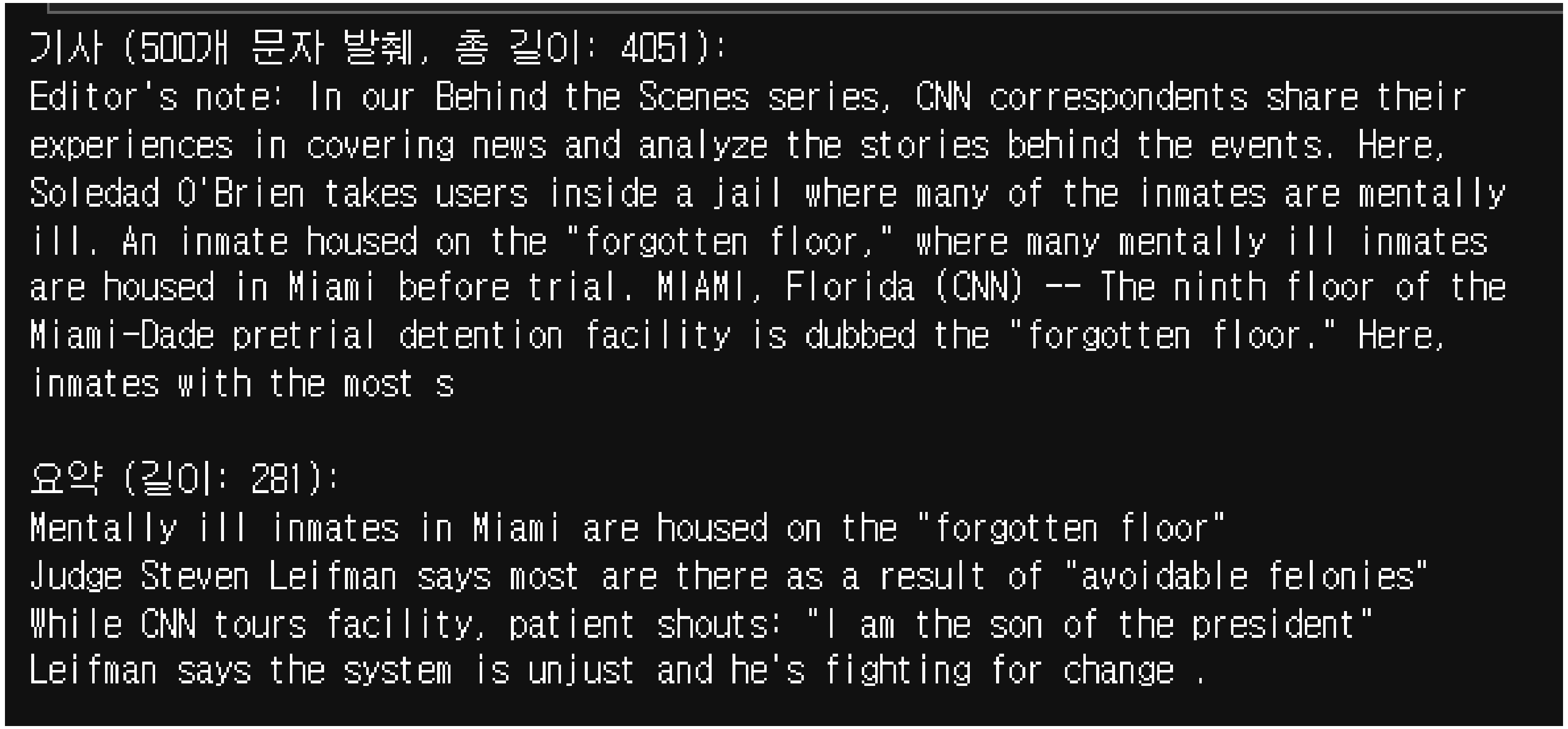


```
"I am sorry, but the server is full. I'm sorry to hear that you're not here to get some coffee. Nothing like that."
"I'm fine, I'm just something you're not allowed to do. I'm not going to ask you to come back to the server. I'm
just here to help you out. I'm just here to help you out." "I'm sorry, it's not like I'm going to ask you to come
back to the server. I'm just here to help you out. I'm just here to help you
```

Application: Summarize



```python
from transformers import pipeline, set_seed
from datasets import load_dataset
sample = dataset['train'][1]

dataset = load_dataset("cnn_dailymail", version="3.0.0")
```



```
기사 (500개 문자 발췌, 총 길이: 4051):
Editor's note: In our Behind the Scenes series, CNN correspondents share their
experiences in covering news and analyze the stories behind the events. Here,
Soledad O'Brien takes users inside a jail where many of the inmates are mentally
ill. An inmate housed on the "forgotten floor," where many mentally ill inmates
are housed in Miami before trial. MIAMI, Florida (CNN) -- The ninth floor of the
Miami-Dade pretrial detention facility is dubbed the "forgotten floor." Here,
inmates with the most s

요약 (길이: 281):
Mentally ill inmates in Miami are housed on the "forgotten floor"
Judge Steven Leifman says most are there as a result of "avoidable felonies"
While CNN tours facility, patient shouts: "I am the son of the president"
Leifman says the system is unjust and he's fighting for change .
```

Application: Summarize

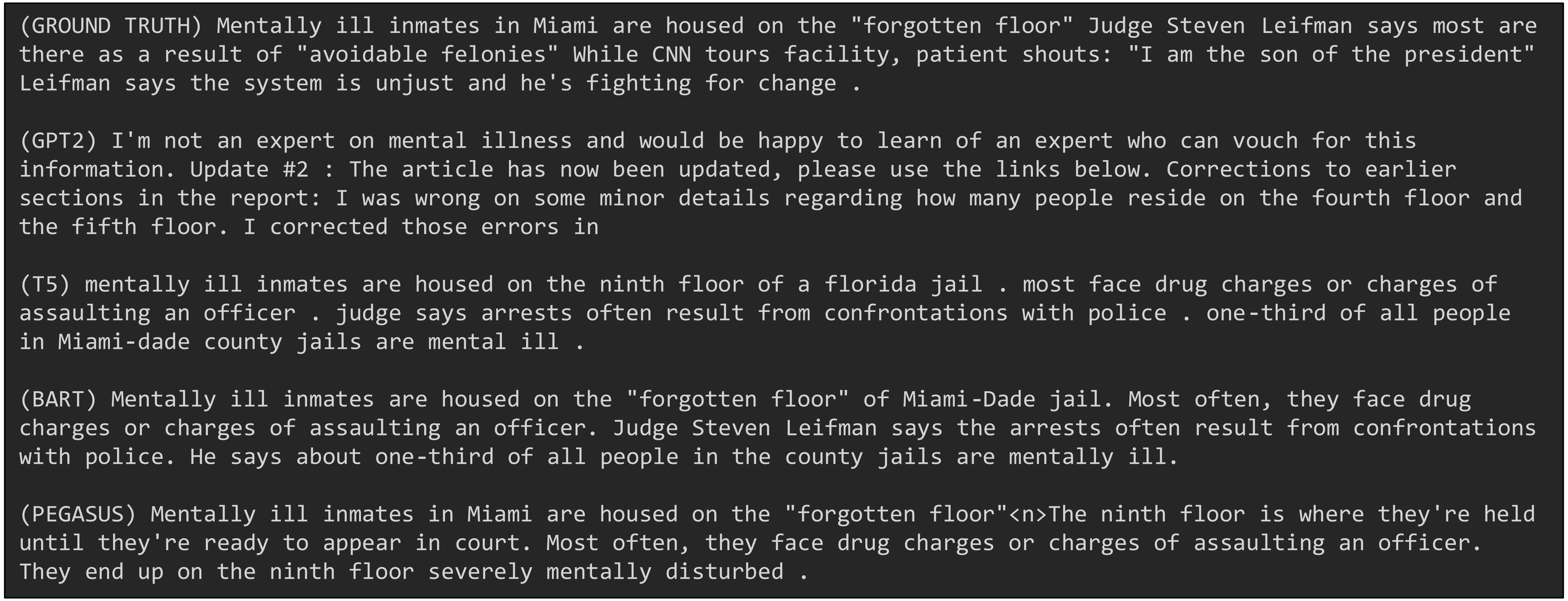
```python
from transformers import pipeline, set_seed

set_seed(42)

pipe = pipeline("text-generation", model="gpt2")

gpt2_query = sample_text + "\nTL;DR:\n"
pipe_out = pipe(gpt2_query, max_length=512, clean_up_tokenization_spaces=True)
summaries["gpt2"] = "\n".join(
    sent_tokenize(pipe_out[0]["generated_text"][len(gpt2_query) :]))
```

```python
pipe = pipeline("summarization", model='t5-large')
pipe_out = pipe(sample_text)
summaries['t5'] = '\n'.join(sent_tokenize(pipe_out[0]['summary_text']))
```

```python
pipe = pipeline("summarization", model='google/pegasus-cnn_dailymail')
pipe_out = pipe(sample_text)
summaries['pegasus'] = '\n'.join(sent_tokenize(pipe_out[0]['summary_text']))
```

```python
pipe = pipeline("summarization", model='google/pegasus-cnn_dailymail')
pipe_out = pipe(sample_text)
summaries['pegasus'] = '\n'.join(sent_tokenize(pipe_out[0]['summary_text']))
```

Application: Summarize



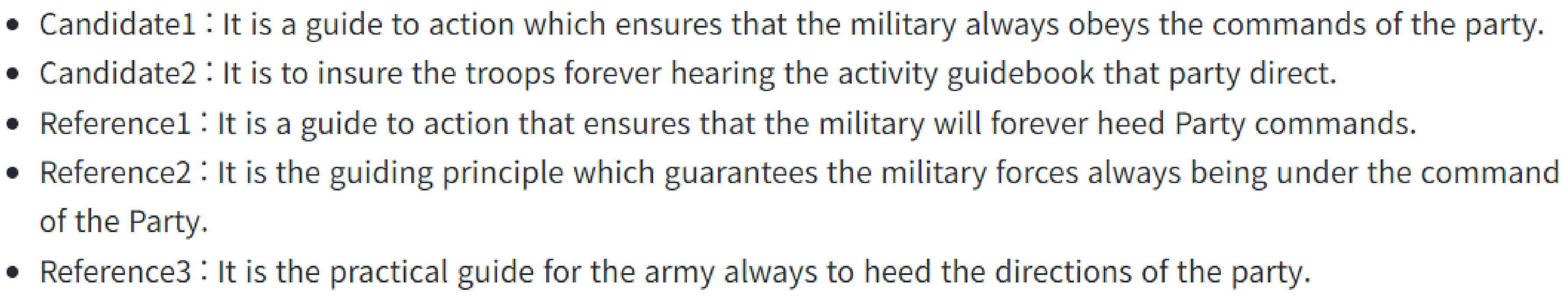
```
(GROUND TRUTH) Mentally ill inmates in Miami are housed on the "forgotten floor" Judge Steven Leifman says most are
there as a result of "avoidable felonies" While CNN tours facility, patient shouts: "I am the son of the president"
Leifman says the system is unjust and he's fighting for change .

(GPT2) I'm not an expert on mental illness and would be happy to learn of an expert who can vouch for this
information. Update #2 : The article has now been updated, please use the links below. Corrections to earlier
sections in the report: I was wrong on some minor details regarding how many people reside on the fourth floor and
the fifth floor. I corrected those errors in

(T5) mentally ill inmates are housed on the ninth floor of a florida jail . most face drug charges or charges of
assaulting an officer . judge says arrests often result from confrontations with police . one-third of all people
in Miami-dade county jails are mental ill .

(BART) Mentally ill inmates are housed on the "forgotten floor" of Miami-Dade jail. Most often, they face drug
charges or charges of assaulting an officer. Judge Steven Leifman says the arrests often result from confrontations
with police. He says about one-third of all people in the county jails are mentally ill.

(PEGASUS) Mentally ill inmates in Miami are housed on the "forgotten floor"<n>The ninth floor is where they're held
until they're ready to appear in court. Most often, they face drug charges or charges of assaulting an officer.
They end up on the ninth floor severely mentally disturbed .
```

그래서 어떤 모델이 좋은가요?

BLEU 평가지표(bilingual evaluation understudy)

- **기존에 사용하던 방식**

$$\text{Unigram Precision} = \frac{\text{Ref들 중에서 존재하는 Ca의 단어의 수}}{\text{Ca의 총 단어 수}}$$



- Candidate1 : It is a guide to action which ensures that the military always obeys the commands of the party.
- Candidate2 : It is to insure the troops forever hearing the activity guidebook that party direct.
- Reference1 : It is a guide to action that ensures that the military will forever heed Party commands.
- Reference2 : It is the guiding principle which guarantees the military forces always being under the command of the Party.
- Reference3 : It is the practical guide for the army always to heed the directions of the party.

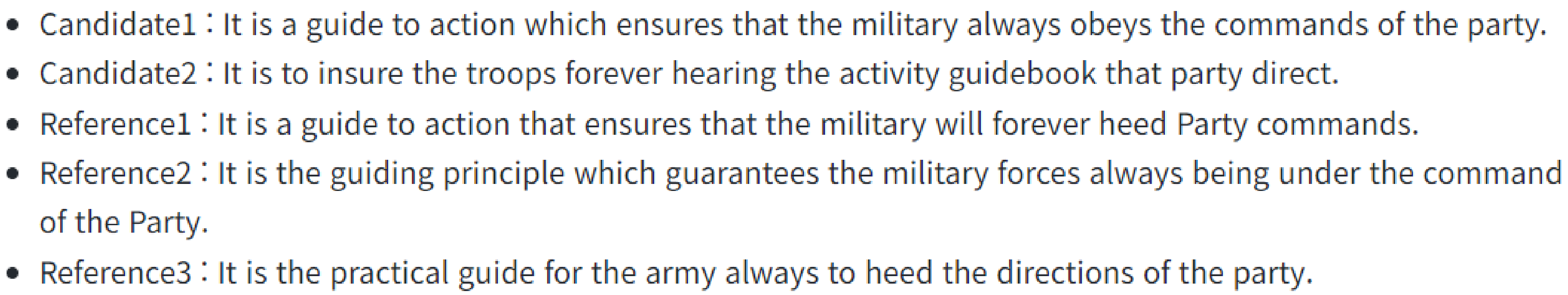
$$\text{Ca1 Unigram Precision} = \frac{17}{18}$$

- 하지만, 이 방식이라면 is is is is is is is is is … is 라는 문장은 정밀도가 1이 된다.

BLEU 평가지표

- **중복을 제거하여 보정된 정밀도**

$$Count_{clip}\ =\ min(Count,\ Max\_Ref\_Count)$$



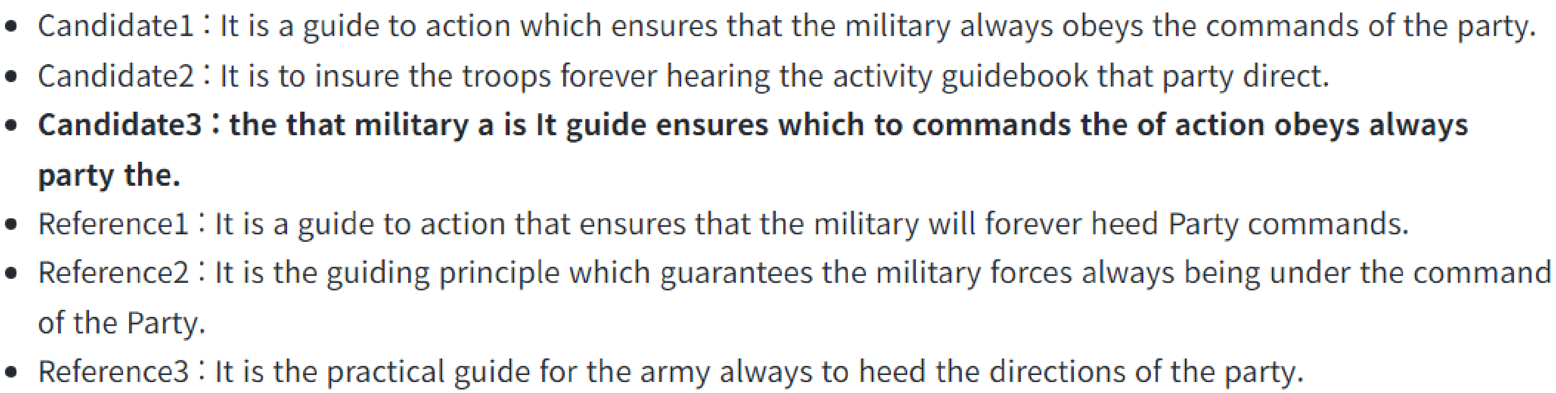
- Candidate1 : It is a guide to action which ensures that the military always obeys the commands of the party.
- Candidate2 : It is to insure the troops forever hearing the activity guidebook that party direct.
- Reference1 : It is a guide to action that ensures that the military will forever heed Party commands.
- Reference2 : It is the guiding principle which guarantees the military forces always being under the command of the Party.
- Reference3 : It is the practical guide for the army always to heed the directions of the party.

각 Ref 문장에서 is는 한번씩만 나왔다. 때문에 is is is is 를 해도 정밀도는 1/n이 된다.

$$\text{Modified Unigram Precision} = \frac{\text{Ca의 각 유니그램에 대해 } Count_{clip}\text{을 수행한 값의 총 합}}{\text{Ca의 총 유니그램 수}} \quad = \frac{\sum_{unigram\in Candidate}\ Count_{clip}(unigram)}{\sum_{unigram\in Candidate}\ Count(unigram)}$$

BLEU 평가지표

- **순서를 고려하여 보정된 정밀도**



- Candidate1 : It is a guide to action which ensures that the military always obeys the commands of the party.
- Candidate2 : It is to insure the troops forever hearing the activity guidebook that party direct.
- **Candidate3 : the that military a is It guide ensures which to commands the of action obeys always party the.**
- Reference1 : It is a guide to action that ensures that the military will forever heed Party commands.
- Reference2 : It is the guiding principle which guarantees the military forces always being under the command of the Party.
- Reference3 : It is the practical guide for the army always to heed the directions of the party.

단순 유니그램으로 하게 되면, Candidate 3은 문법이 이상하지만 정밀도 0.944가 나온다.

때문에 bleu에서는 n-gram을 섞어서 사용하게 된다.

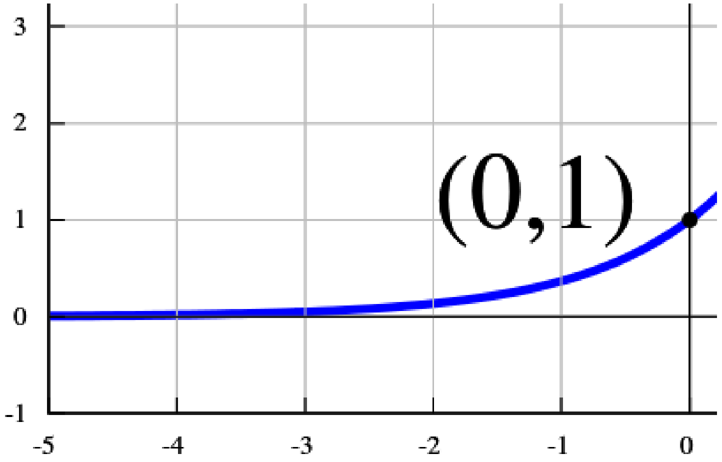
$$\left(\prod_{n=1}^{N} p_n\right)^{1/N}$$

BLEU

- **Brevity penalty**

$$BR = min(1, e^{1-\ell_{\text{ref}}/\ell_{\text{gen}}})$$

$$\text{BLEU-}N = BR \times \left(\prod_{n=1}^{N} p_n\right)^{1/N}$$



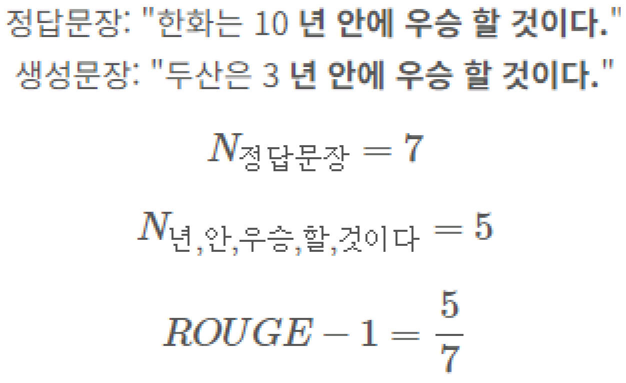
ref와 gen 을 비교해서 gen의 길이가 길면 1이 된다.

gen의 길이가 짧으면, 지수 함수를 따라간다.

Rouge 평가지표 (Recall-Oriented Understudy for Gisting Evaluation)

- **Recall 값 고려하기**

$$\text{ROUGE-}N=\frac{\sum_{snt'\in c}\sum_{n-gram\in snt'} Count_{match}(n\text{-}gram)}{\sum_{snt'\in c}\sum_{n-gram\in snt'} Count(n\text{-}gram)}$$



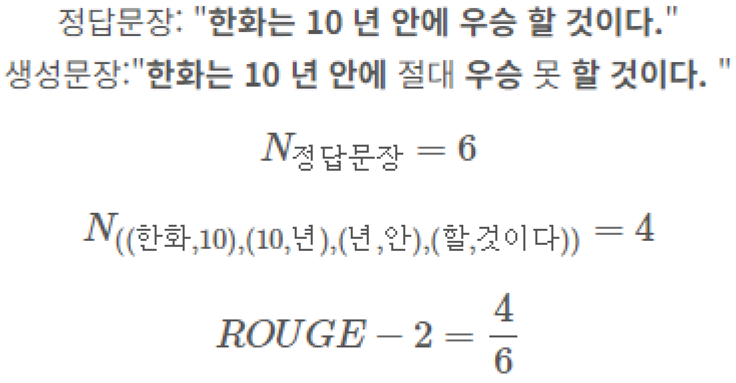
정답문장: "한화는 10 **년 안에 우승 할 것이다.**"

생성문장: "두산은 3 **년 안에 우승 할 것이다.**"

$$N_{정답문장} = 7$$

$$N_{년,안,우승,할,것이다} = 5$$

$$ROUGE-1 = \frac{5}{7}$$



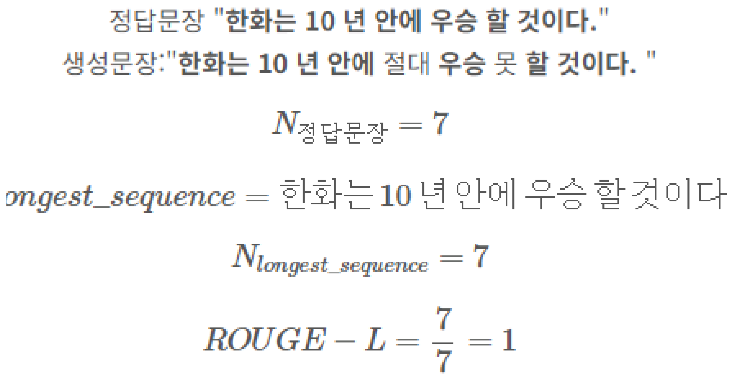
정답문장: "**한화는 10 년 안에 우승 할 것이다.**"

생성문장:"**한화는 10 년 안에** 절대 **우승** 못 **할 것이다.** "

$$N_{정답문장} = 6$$

$$N_{((한화,10),(10,년),(년,안),(할,것이다))} = 4$$

$$ROUGE-2 = \frac{4}{6}$$



정답문장 "**한화는 10 년 안에 우승 할 것이다.**"

생성문장:"**한화는 10 년 안에** 절대 **우승** 못 **할 것이다.** "

$$N_{정답문장} = 7$$

$$ongest\_sequence = 한화는 10 년 안에 우승 할것이다$$

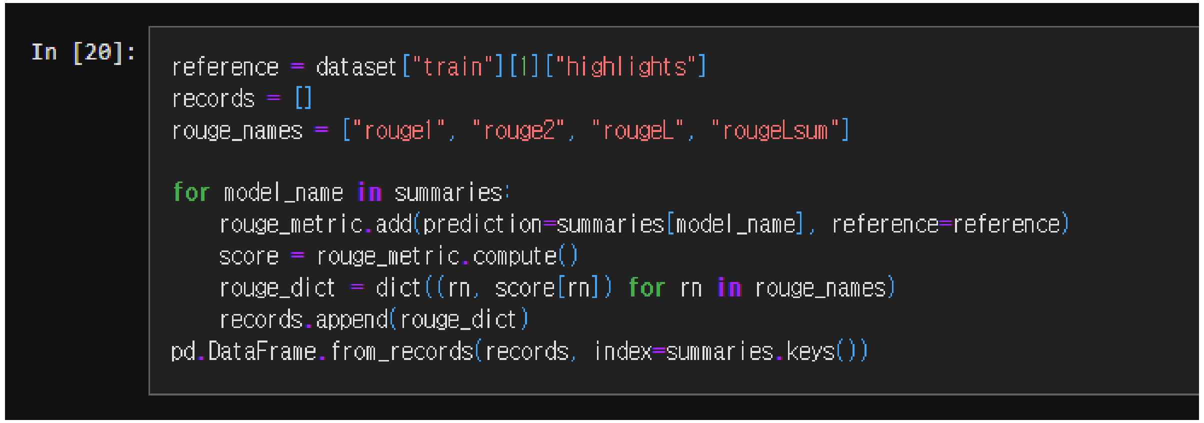
$$N_{longest\_sequence} = 7$$

$$ROUGE-L = \frac{7}{7} = 1$$

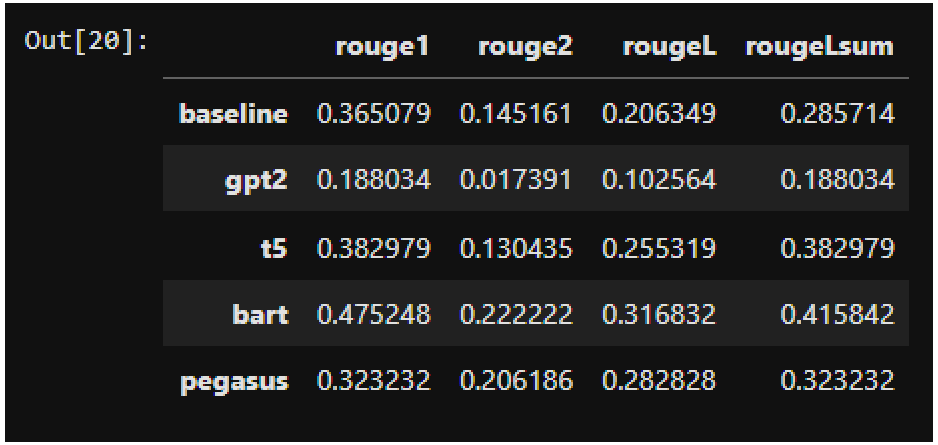
참고 블로그: https://supkoon.tistory.com/26

적용해보기

- **ROUGE**



```
In [20]:
reference = dataset["train"][1]["highlights"]
records = []
rouge_names = ["rouge1", "rouge2", "rougeL", "rougeLsum"]

for model_name in summaries:
    rouge_metric.add(prediction=summaries[model_name], reference=reference)
    score = rouge_metric.compute()
    rouge_dict = dict((rn, score[rn]) for rn in rouge_names)
    records.append(rouge_dict)
pd.DataFrame.from_records(records, index=summaries.keys())
```



Out[20]:

	rouge1	rouge2	rougeL	rougeLsum
baseline	0.365079	0.145161	0.206349	0.285714
gpt2	0.188034	0.017391	0.102564	0.188034
t5	0.382979	0.130435	0.255319	0.382979
bart	0.475248	0.222222	0.316832	0.415842
pegasus	0.323232	0.206186	0.282828	0.323232

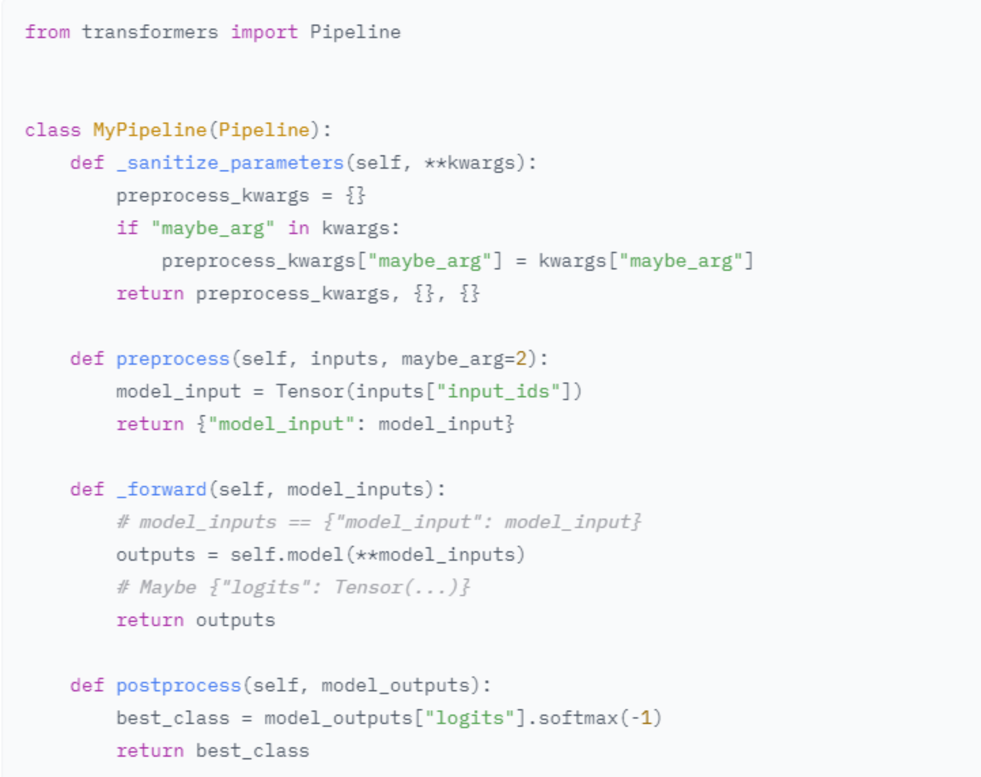
Conclusion

1. 빔 서치 방식과 같이 문장의 표현을 보는 것이 좋으나, 계산상의 이유로 그리디 서치 방식을 사용하는 것이 좋을 수도 있다.

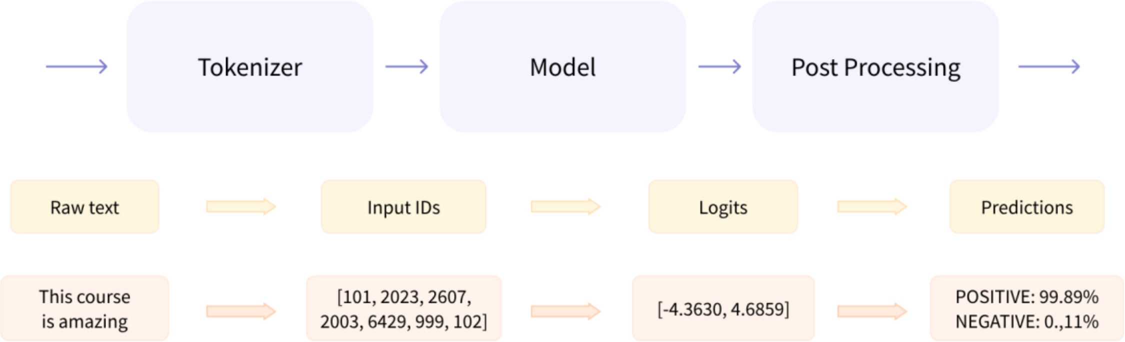
2. 샘플링 방법으로 온도상수를 이용하거나 top-k 단어를 뽑을 수 있다.

3. 평가 방법으로 bleu나 rouge를 이용할 수 있다.

Appendix



```python
from transformers import Pipeline


class MyPipeline(Pipeline):
    def _sanitize_parameters(self, **kwargs):
        preprocess_kwargs = {}
        if "maybe_arg" in kwargs:
            preprocess_kwargs["maybe_arg"] = kwargs["maybe_arg"]
        return preprocess_kwargs, {}, {}

    def preprocess(self, inputs, maybe_arg=2):
        model_input = Tensor(inputs["input_ids"])
        return {"model_input": model_input}

    def _forward(self, model_inputs):
        # model_inputs == {"model_input": model_input}
        outputs = self.model(**model_inputs)
        # Maybe {"logits": Tensor(...)}
        return outputs

    def postprocess(self, model_outputs):
        best_class = model_outputs["logits"].softmax(-1)
        return best_class
```



Tokenizer

Model

Post Processing

Raw text

Input IDs

Logits

Predictions

This course
is amazing

[101, 2023, 2607,
2003, 6429, 999, 102]

[-4.3630, 4.6859]

POSITIVE: 99.89%
NEGATIVE: 0.,11%