Text Generation
Natural Language Generation (NLG)

1102AITA08
MBA, IM, NTPU (M5026) (Spring 2022)
Tue 2, 3, 4 (9:10-12:00) (B8F40)

https://meet.google.com/paj-zhhj-mya

Min-Yuh Day, Ph.D,
Associate Professor
Institute of Information Management, National Taipei University
https://web.ntpu.edu.tw/~myday

2022-05-03

Syllabus

Week	Date	Subject/Topics
1	2022/02/22	Introduction to Artificial Intelligence for Text Analytics
2	2022/03/01	Foundations of Text Analytics: Natural Language Processing (NLP)
3	2022/03/08	Python for Natural Language Processing
4	2022/03/15	Natural Language Processing with Transformers
5	2022/03/22	Case Study on Artificial Intelligence for Text Analytics I
6	2022/03/29	Text Classification and Sentiment Analysis

Syllabus

Week	Date	Subject/Topics
7	2022/04/05	Tomb-Sweeping Day (Holiday, No Classes)
8	2022/04/12	Midterm Project Report
9	2022/04/19	Multilingual Named Entity Recognition (NER), Text Similarity and Clustering
10	2022/04/26	Text Summarization and Topic Models
11	2022/05/03	Text Generation
12	2022/05/10	Case Study on Artificial Intelligence for Text Analytics II

Syllabus

Week	Date	Subject/Topics
13	2022/05/17	Question Answering and Dialogue Systems
14	2022/05/24	Deep Learning, Transfer Learning, Zero-Shot, and Few-Shot Learning for Text Analytics
15	2022/05/31	Final Project Report I
16	2022/06/07	Final Project Report II
17	2022/06/14	Self-learning
18	2022/06/21	Self-learning

Text Generation

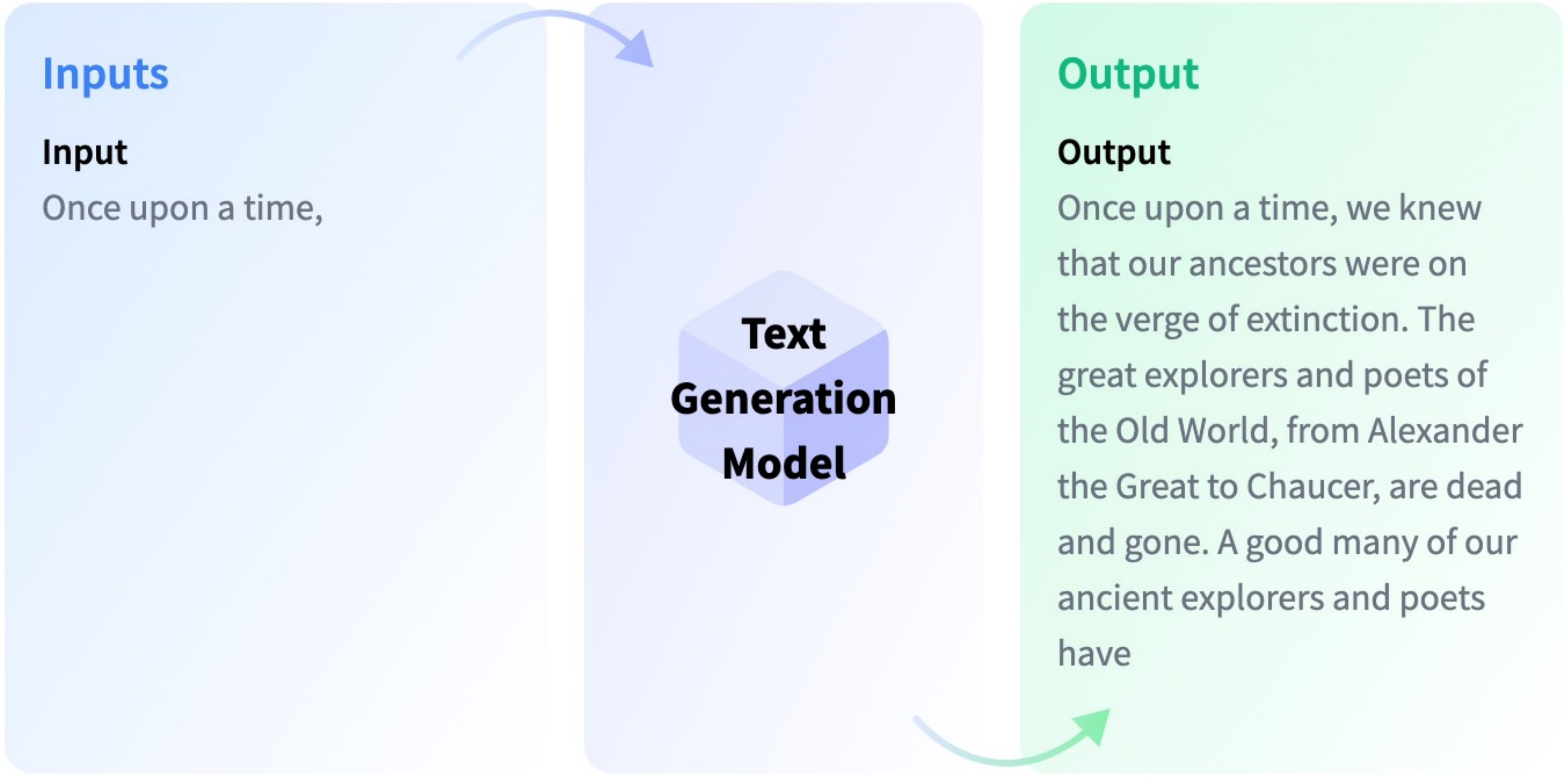
Outline

- **Text Generation**
  - **Natural Language Generation (NLG)**
    - Language Modeling
    - Conditional Language Modeling
  - **Next Word Prediction**
- **Decoding Algorithm**
  - **Greedy Search Decoding**
  - **Beam Search Decoding**
  - **Sampling Methods**
  - **Top-k and Nucleus Sampling**

Text Generation

- Natural Language Generation (NLG)
  - Language Modeling
  - Conditional Language Modeling
- Next Word Prediction

Text Generation



Inputs

Input

Once upon a time,

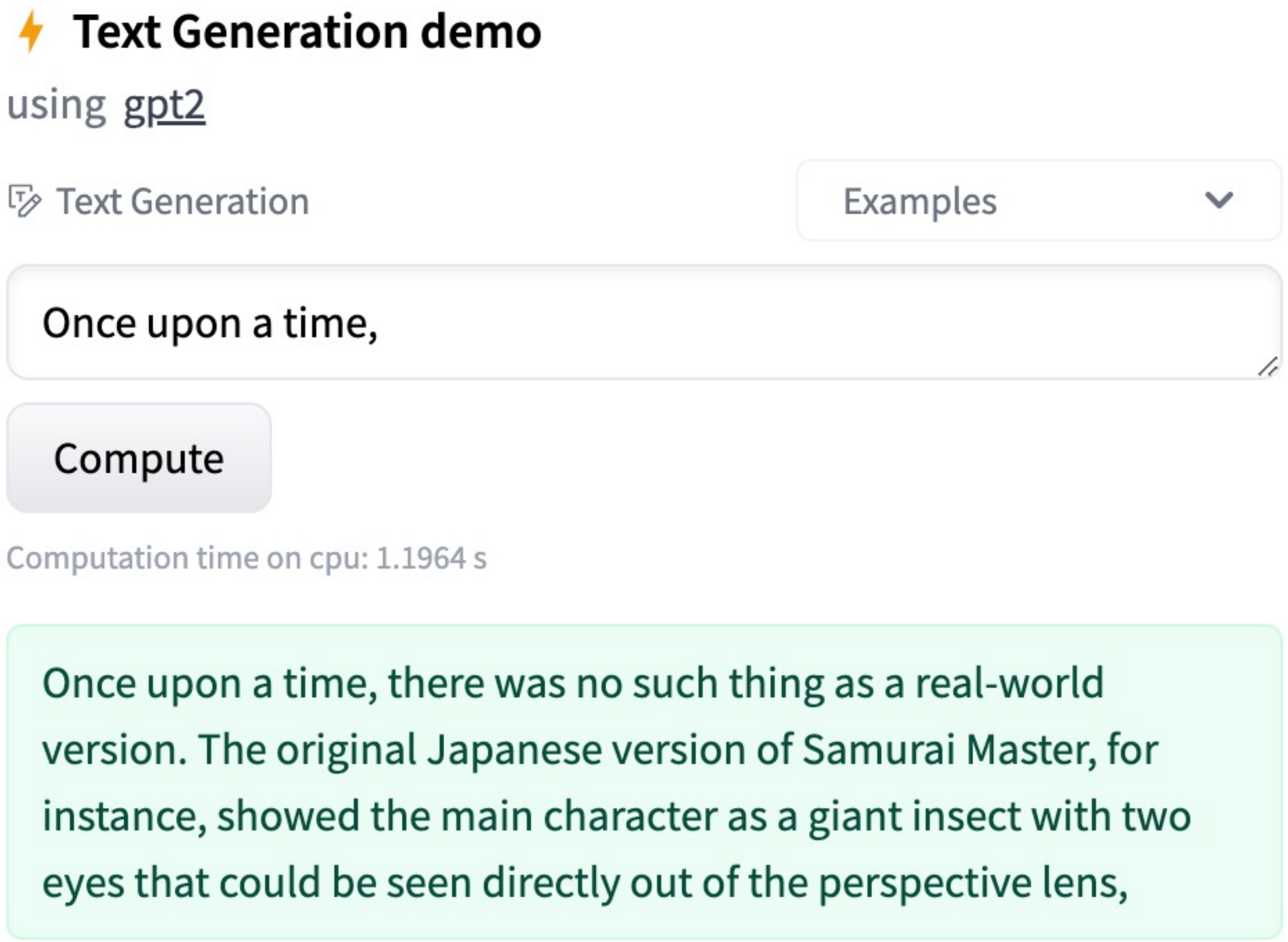
Output

Output

Once upon a time, we knew that our ancestors were on the verge of extinction. The great explorers and poets of the Old World, from Alexander the Great to Chaucer, are dead and gone. A good many of our ancient explorers and poets have

https://huggingface.co/tasks/text-generation

Text Generation



Once upon a time,

Computation time on cpu: 1.1964 s

Once upon a time, there was no such thing as a real-world version. The original Japanese version of Samurai Master, for instance, showed the main character as a giant insect with two eyes that could be seen directly out of the perspective lens,

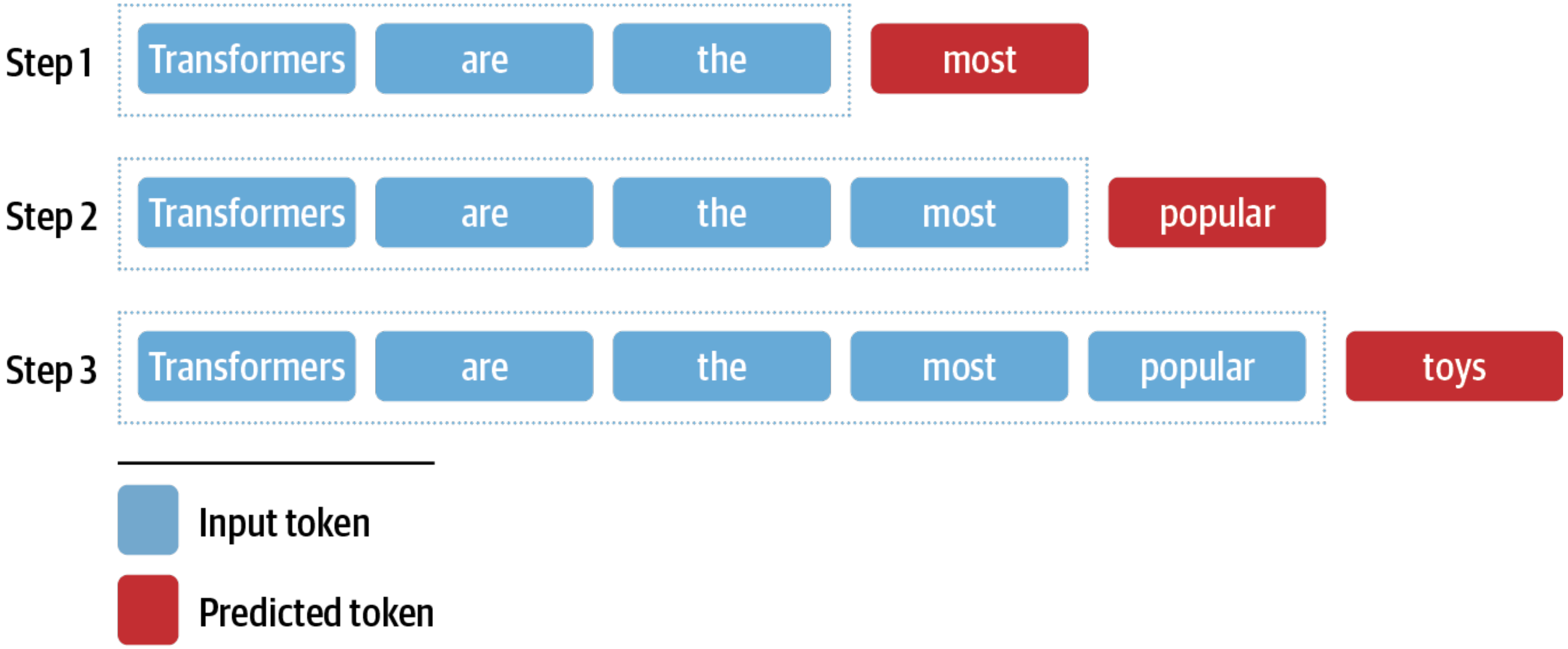
https://huggingface.co/tasks/text-generation

Text Generation

Natural Language Generation (NLG)

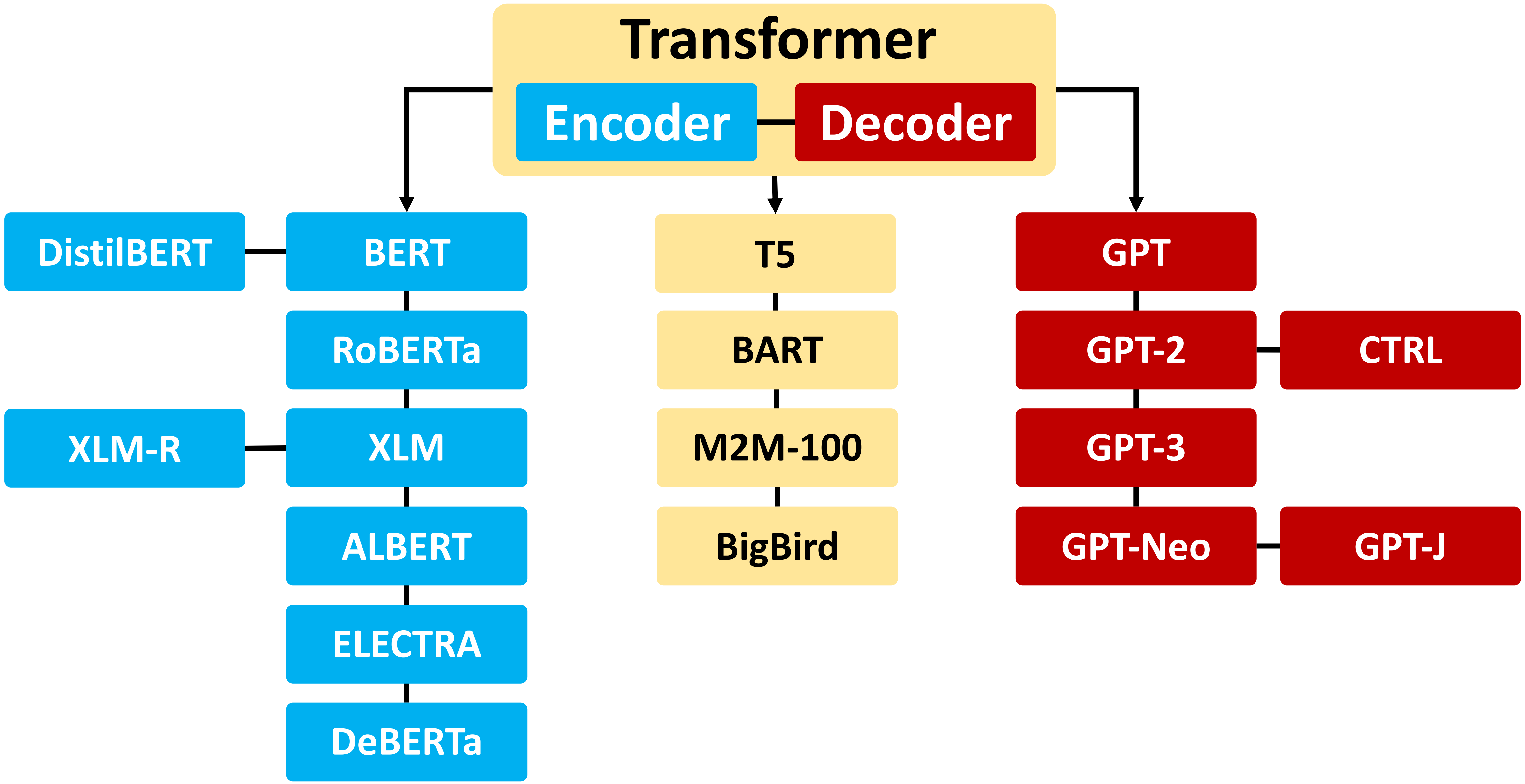
The Challenge with Generating Coherent Text

Next Word Prediction



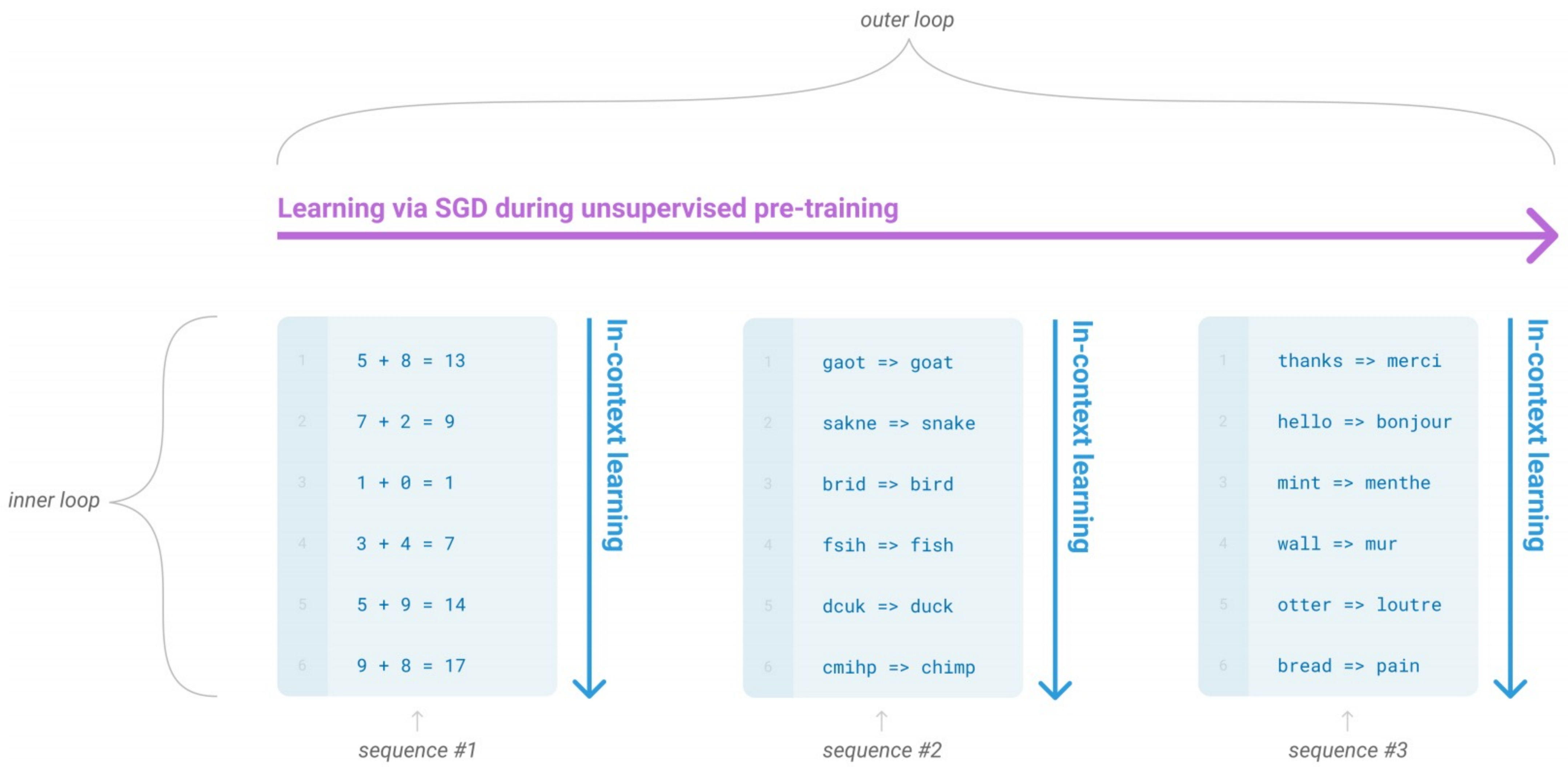
Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.

Transformer Models



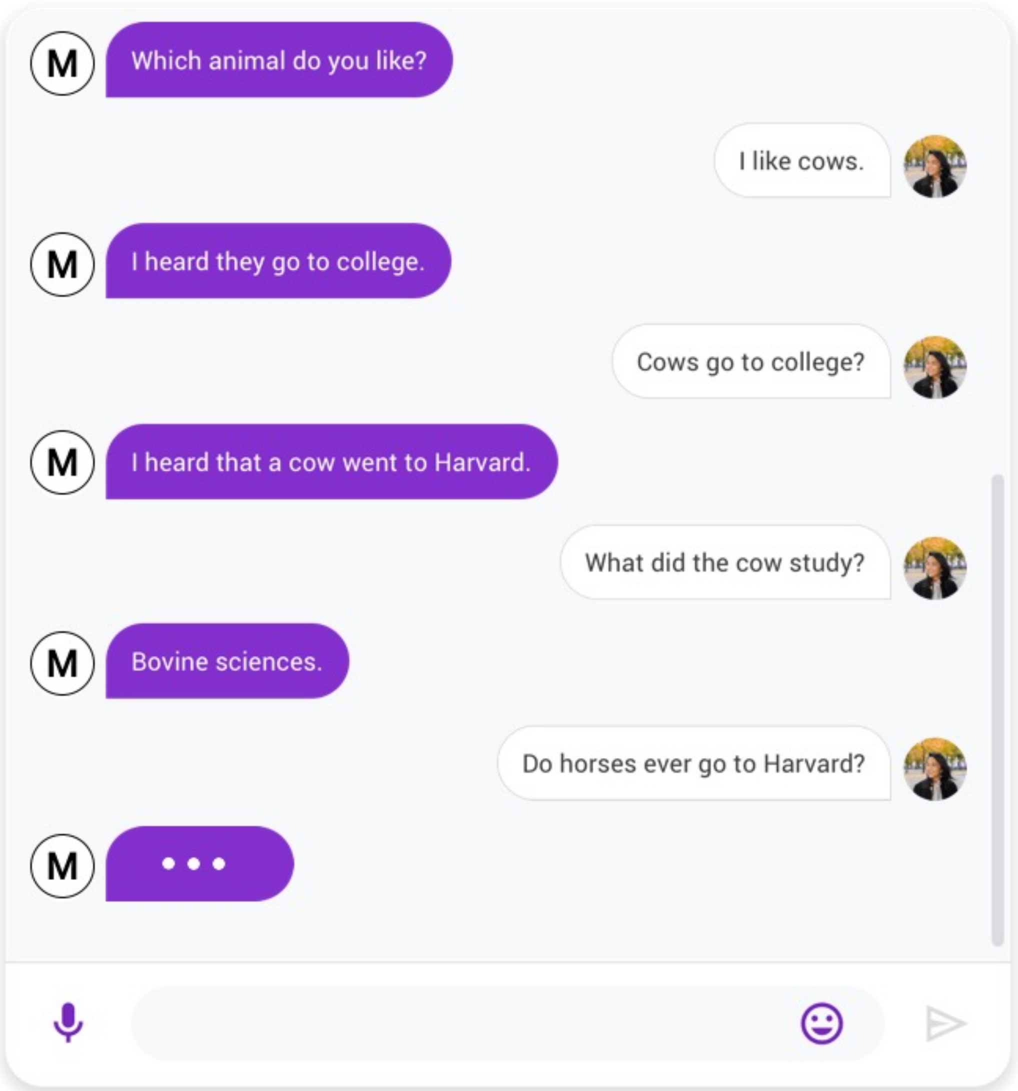
Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.

Text Generation



Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.

Text Generation



Which animal do you like?

I like cows.

I heard they go to college.

Cows go to college?

I heard that a cow went to Harvard.

What did the cow study?

Bovine sciences.

Do horses ever go to Harvard?

• • •

Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.

Text Generation

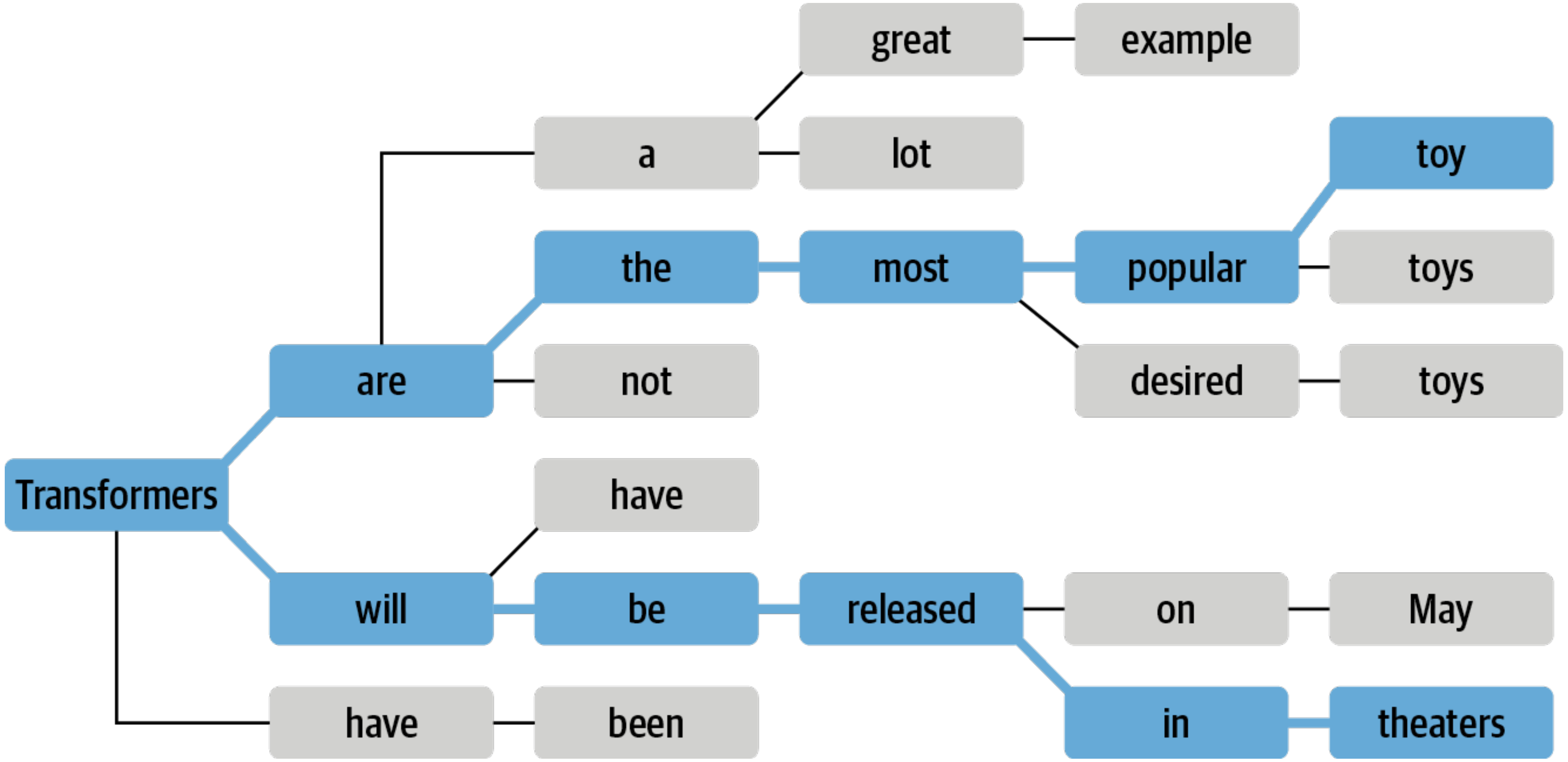
Decoding Algorithm

- Greedy Search Decoding
- Beam Search Decoding
- Sampling Methods
- Top-k and Nucleus Sampling

Text Generation
Greedy Search Decoding

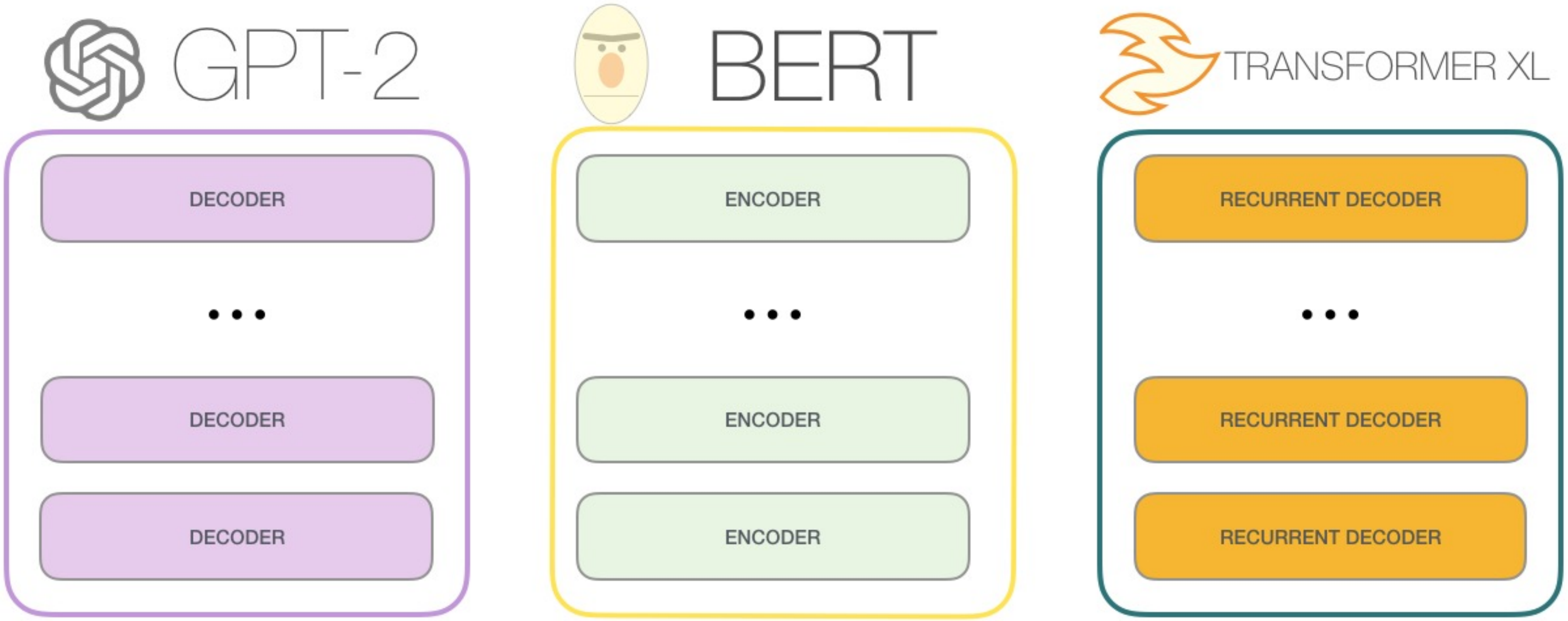
	Input	Choice 1	Choice 2	Choice 3	Choice 4	Choice 5
0	Transformers are the	most (8.53%)	only (4.96%)	best (4.65%)	Transformers (4.37%)	ultimate (2.16%)
1	Transformers are the most	popular (16.78%)	powerful (5.37%)	common (4.96%)	famous (3.72%)	successful (3.20%)
2	Transformers are the most popular	toy (10.63%)	toys (7.23%)	Transformers (6.60%)	of (5.46%)	and (3.76%)
3	Transformers are the most popular toy	line (34.38%)	in (18.20%)	of (11.71%)	brand (6.10%)	line (2.69%)
4	Transformers are the most popular toy line	in (46.28%)	of (15.09%)	, (4.94%)	on (4.40%)	ever (2.72%)
5	Transformers are the most popular toy line in	the (65.99%)	history (12.42%)	America (6.91%)	Japan (2.44%)	North (1.40%)
6	Transformers are the most popular toy line in the	world (69.26%)	United (4.55%)	history (4.29%)	US (4.23%)	U (2.30%)
7	Transformers are the most popular toy line in the world	, (39.73%)	. (30.64%)	and (9.87%)	with (2.32%)	today (1.74%)

Text Generation
Beam Search Decoding

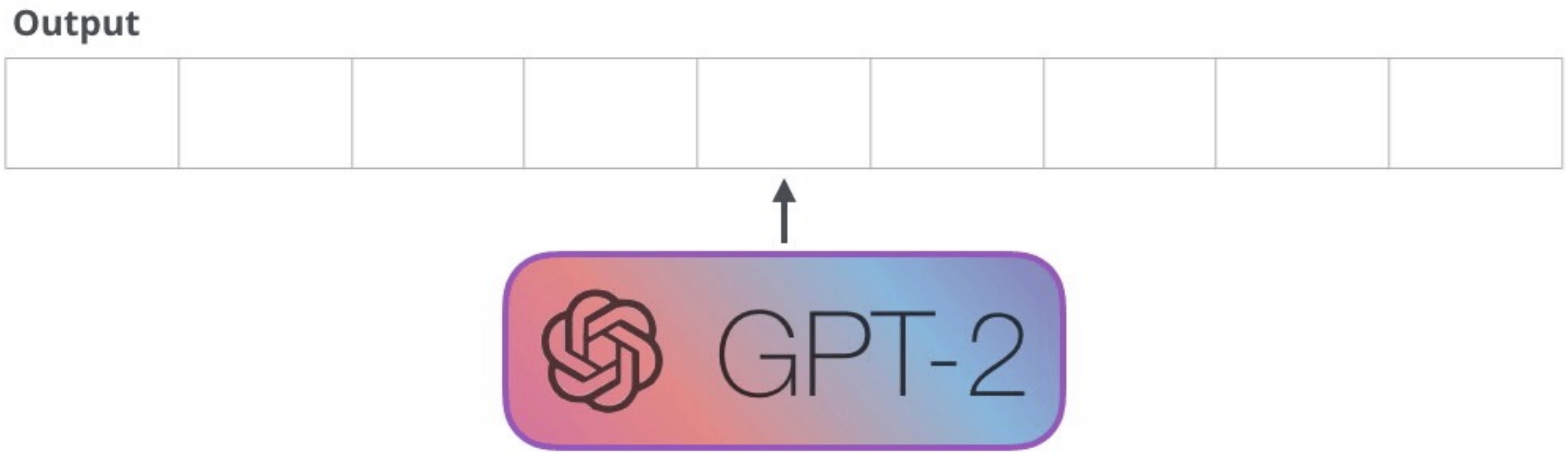


The Illustrated GPT-2
(Visualizing Transformer Language Models)

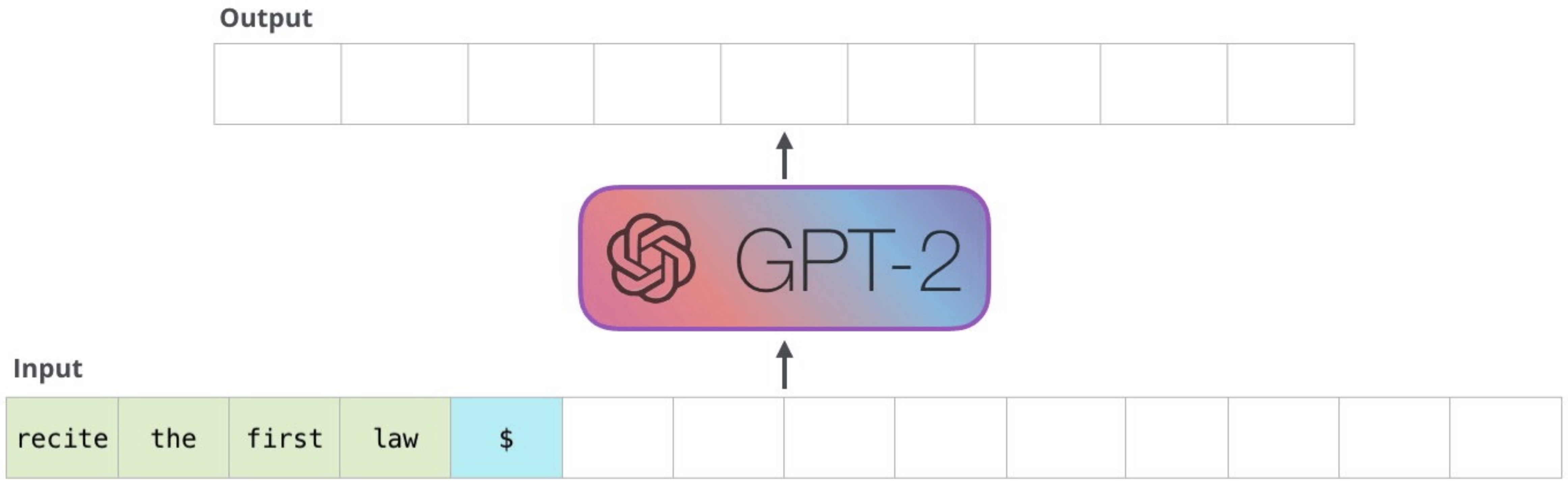
Jay Alammar (2019)



GPT-2 Output

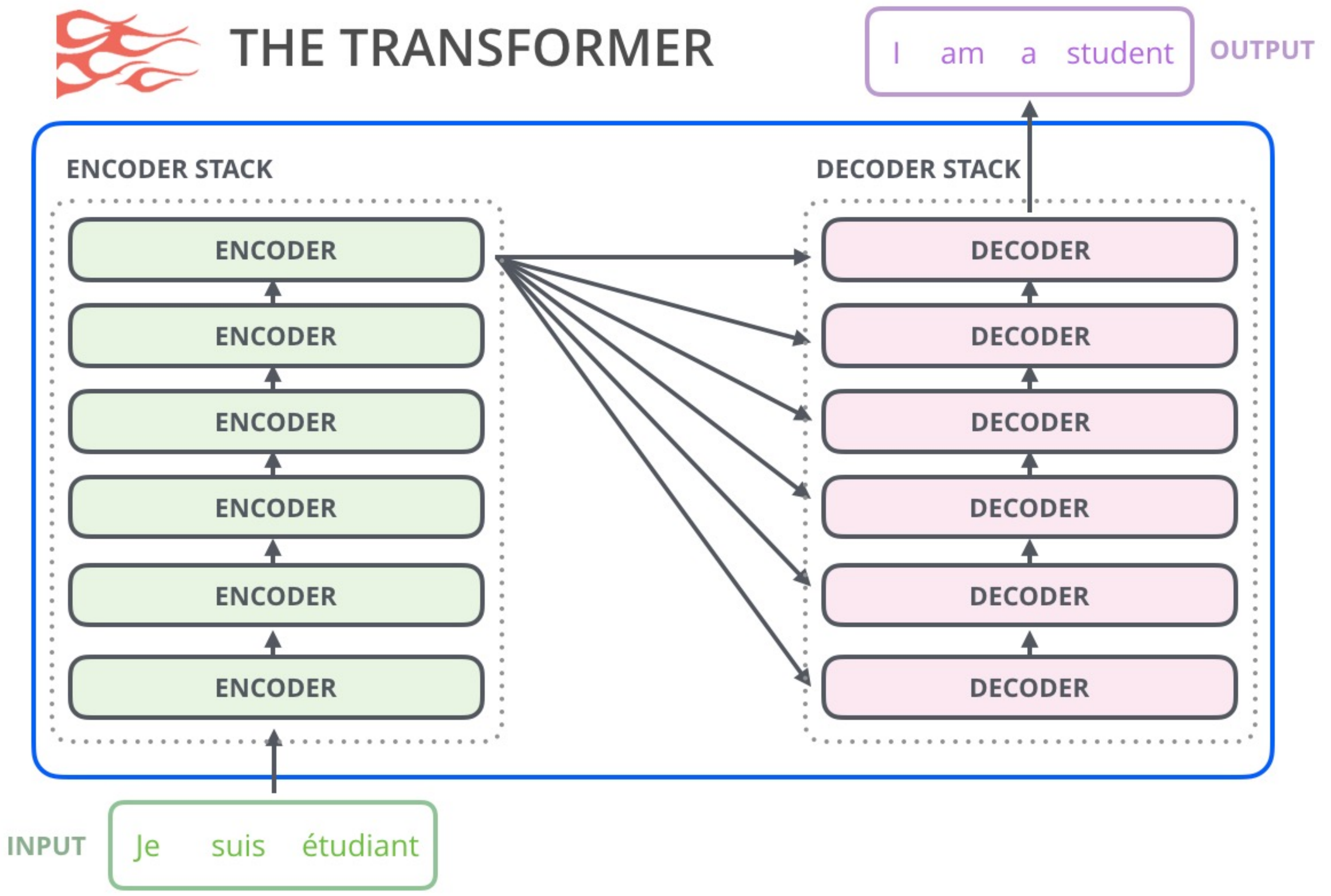


GPT-2 Autoregression



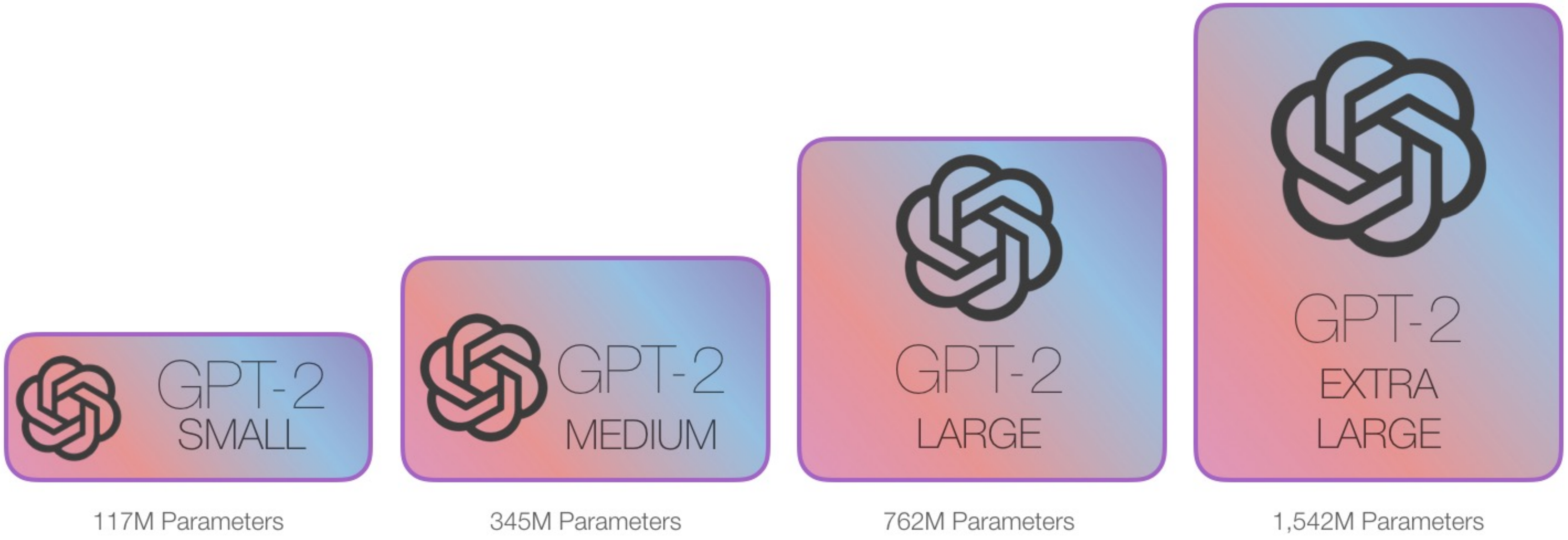
Source: Jay Alammar (2019), The Illustrated GPT-2 (Visualizing Transformer Language Models), https://jalammar.github.io/illustrated-gpt2/

Transformer Encoder Decoder



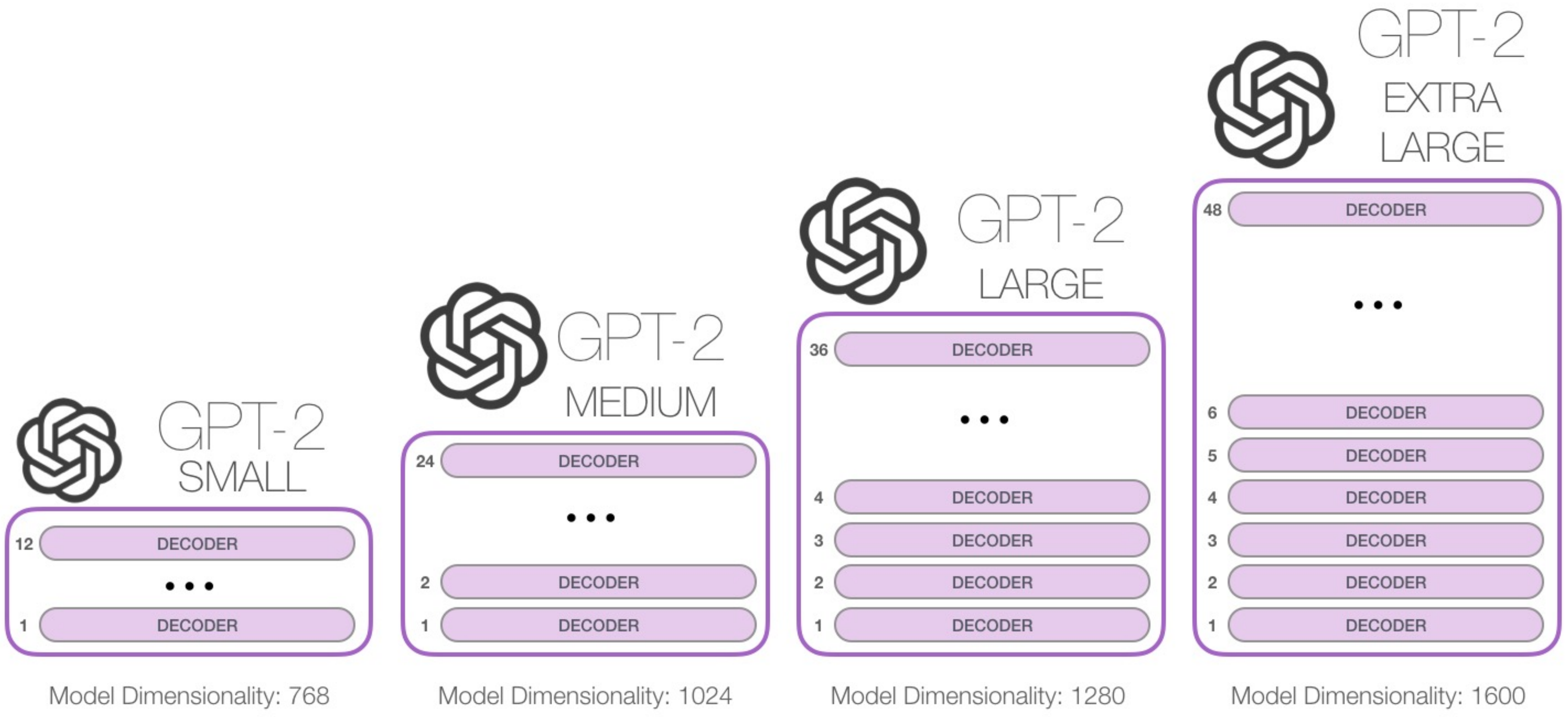
Source: Jay Alammar (2019), The Illustrated GPT-2 (Visualizing Transformer Language Models), https://jalammar.github.io/illustrated-gpt2/

GPT-2 Sizes



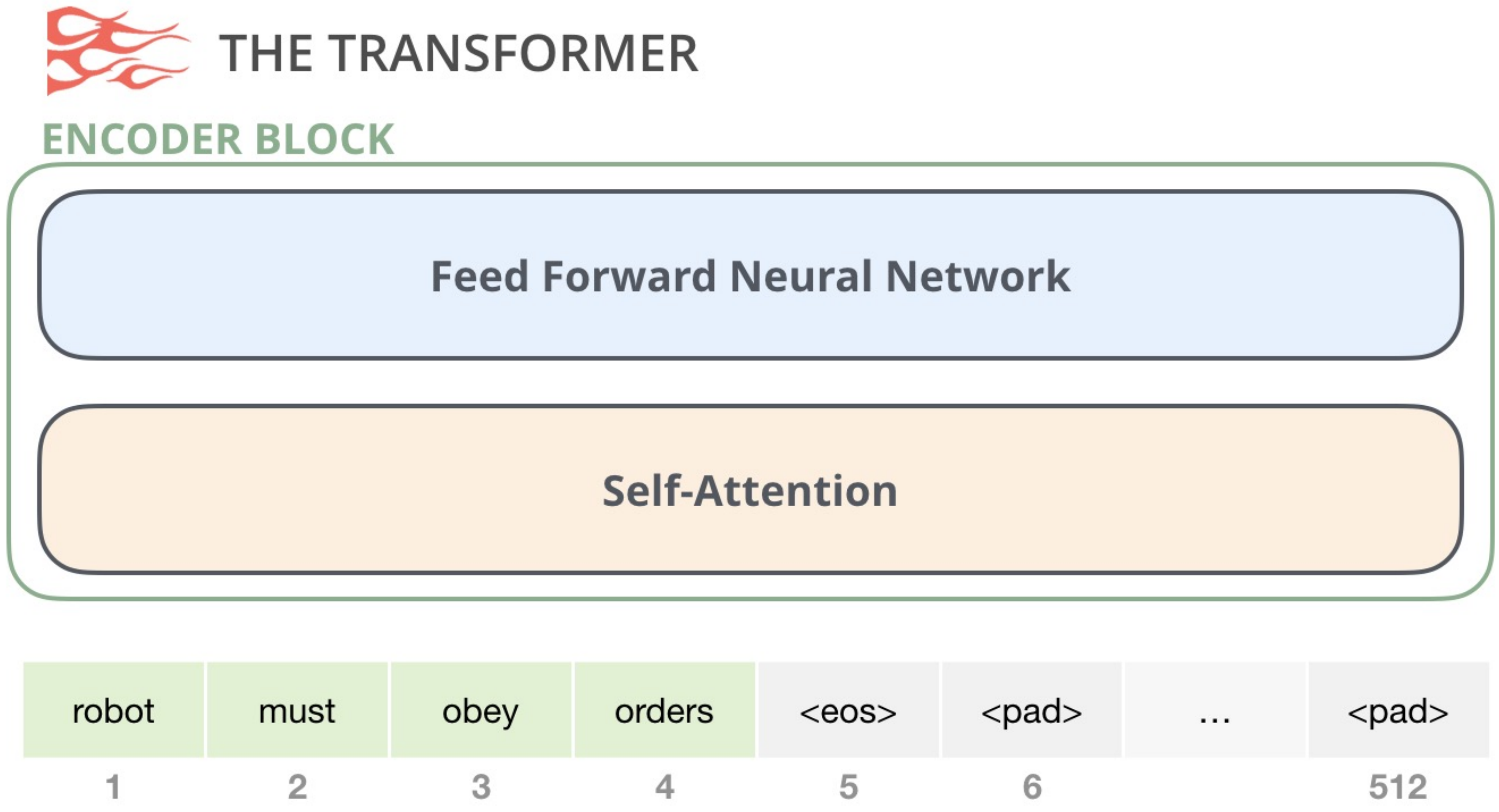
Source: Jay Alammar (2019), The Illustrated GPT-2 (Visualizing Transformer Language Models), https://jalammar.github.io/illustrated-gpt2/

GPT-2 Sizes Hyperparameters



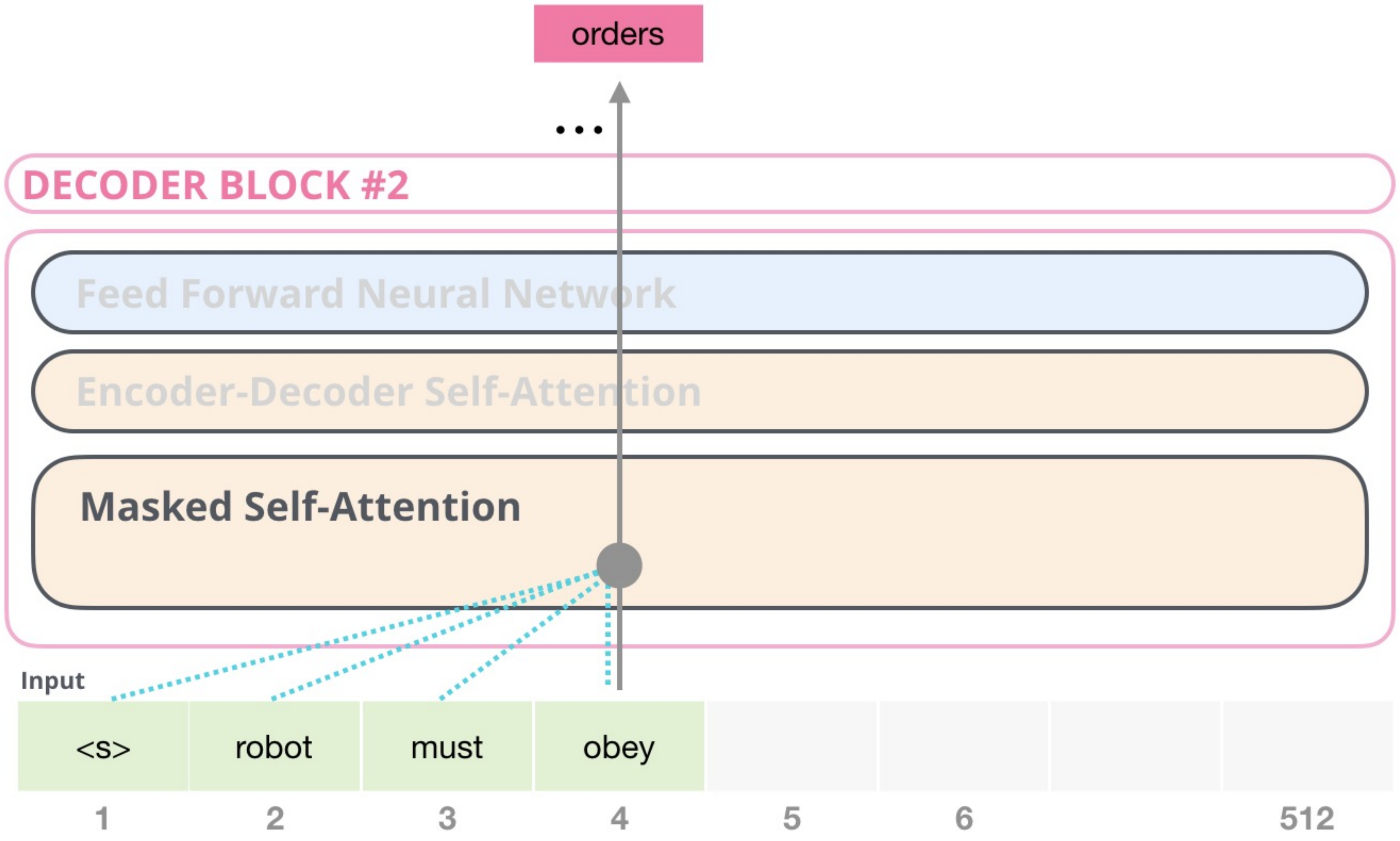
Source: Jay Alammar (2019), The Illustrated GPT-2 (Visualizing Transformer Language Models), https://jalammar.github.io/illustrated-gpt2/

Transformer Encoder

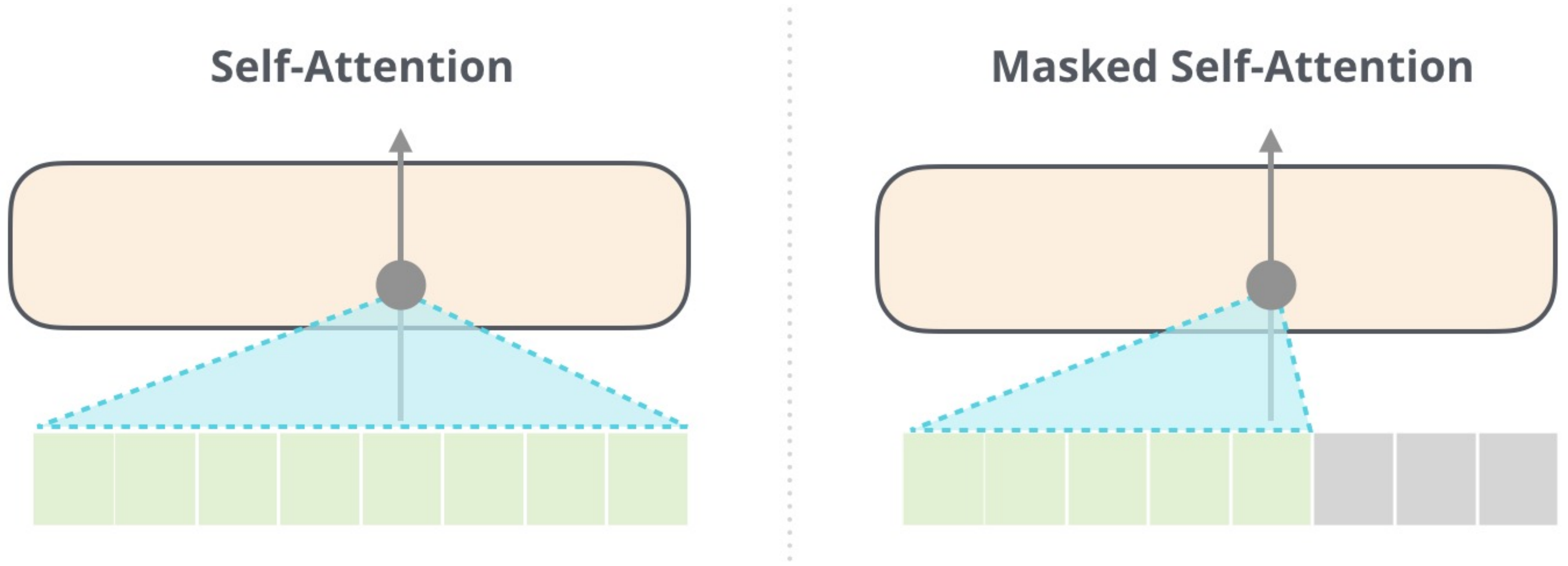


Source: Jay Alammar (2019), The Illustrated GPT-2 (Visualizing Transformer Language Models), https://jalammar.github.io/illustrated-gpt2/

Transformer Decoder

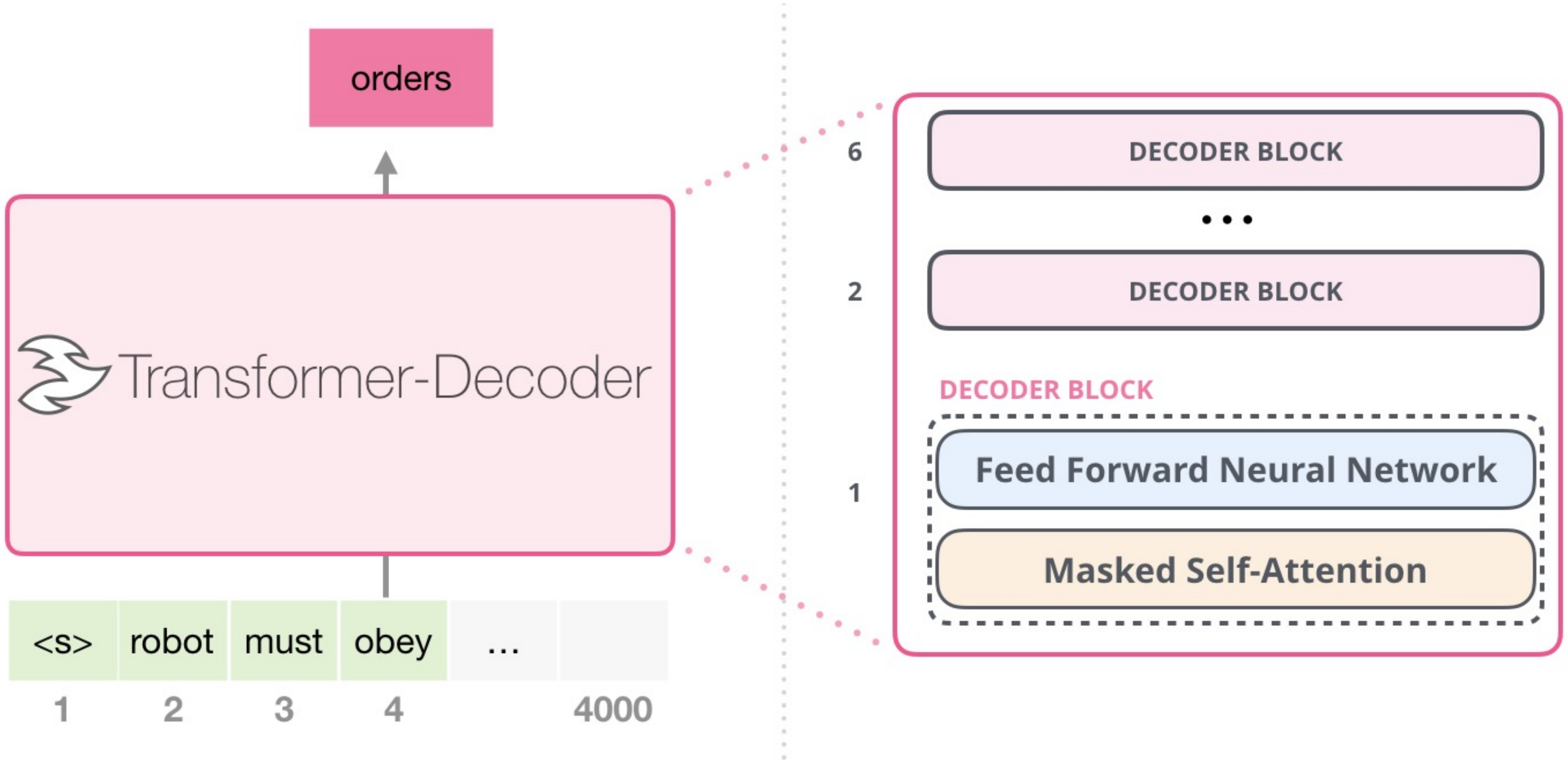


Self-Attention and Masked-Self-Attention

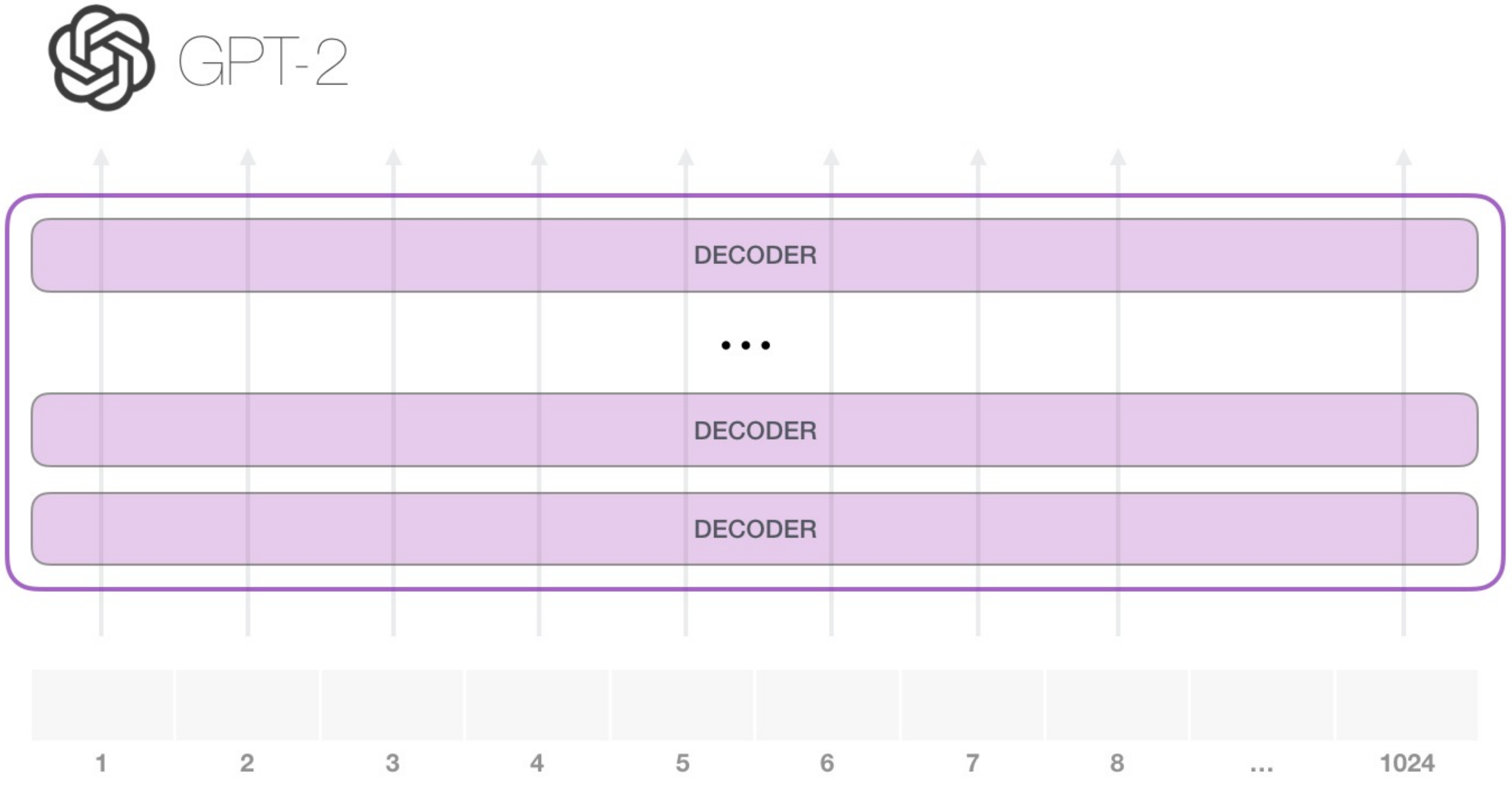


Source: Jay Alammar (2019), The Illustrated GPT-2 (Visualizing Transformer Language Models), https://jalammar.github.io/illustrated-gpt2/

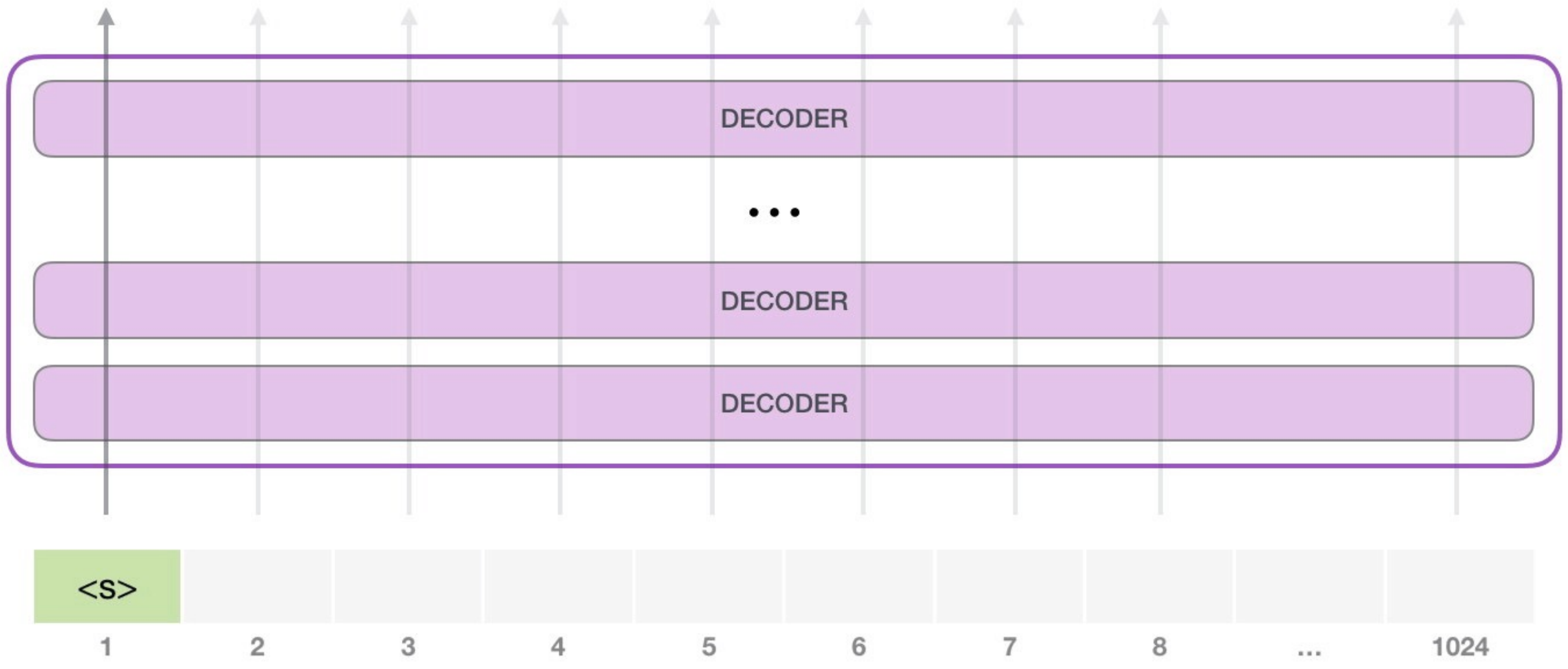
Transformer Decoder



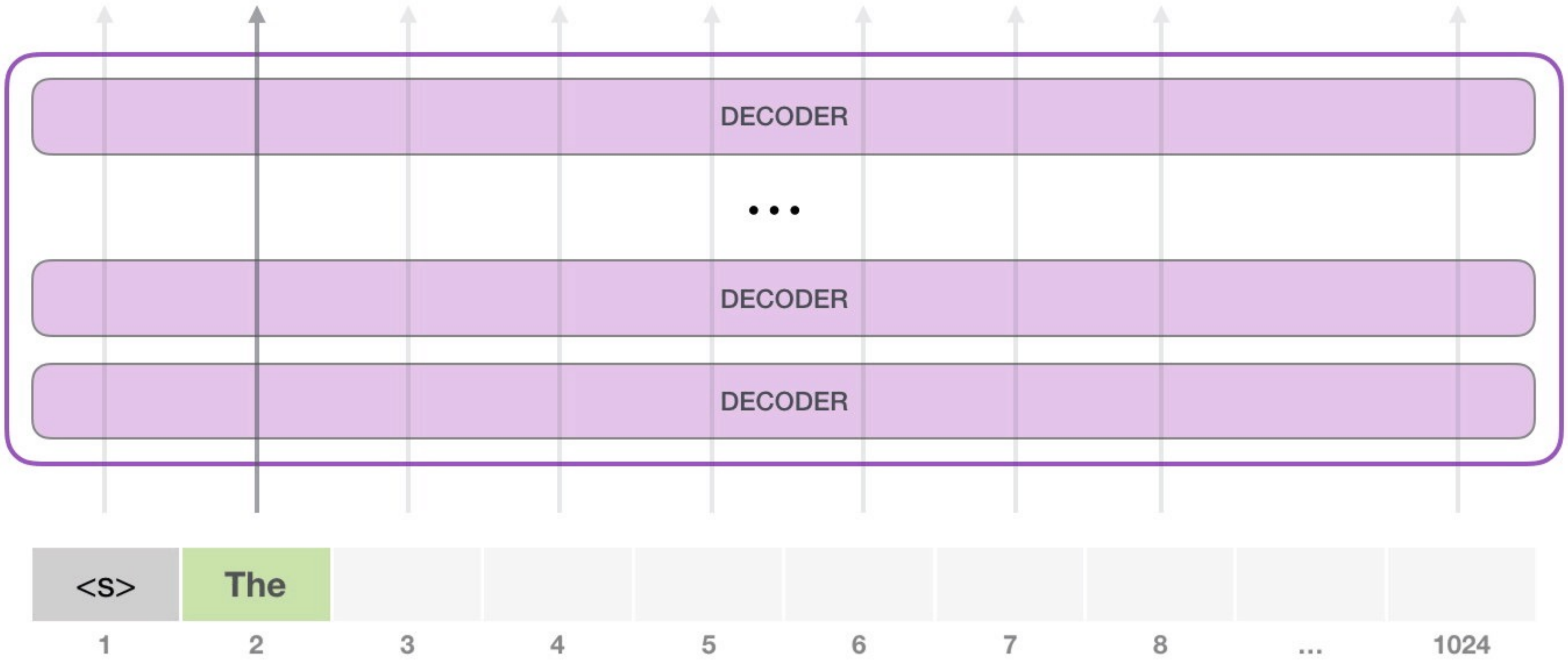
GPT-2 Layers



GPT-2 Simple Ooutput



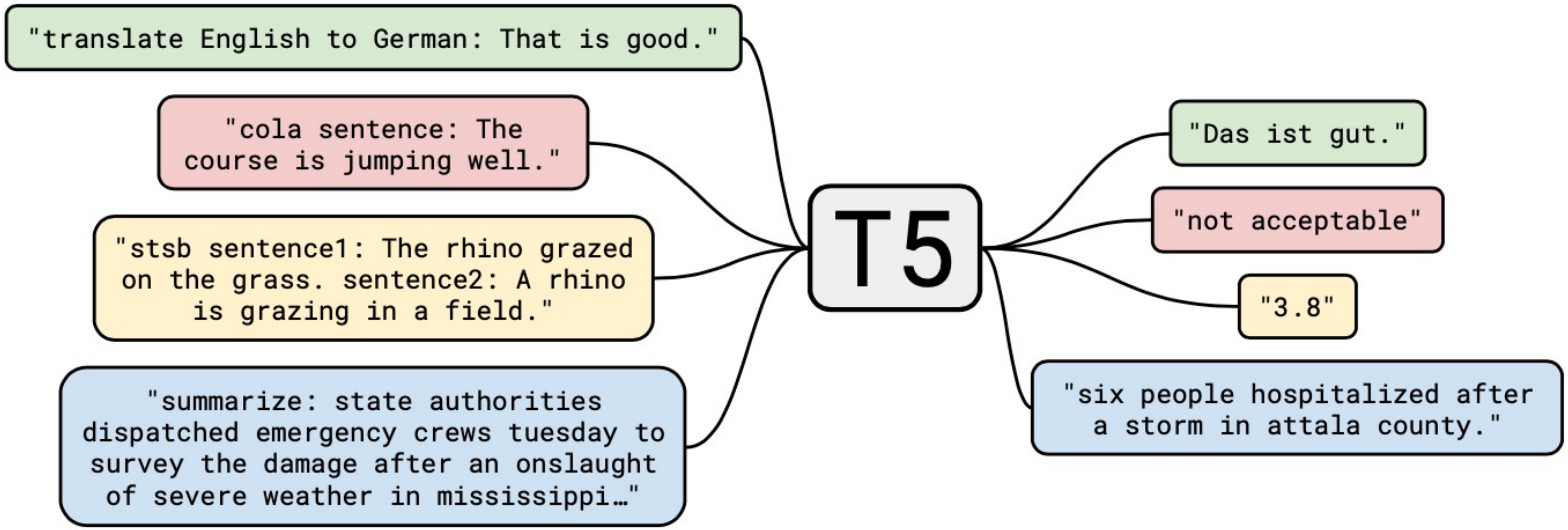
GPT-2 Simple Ooutput



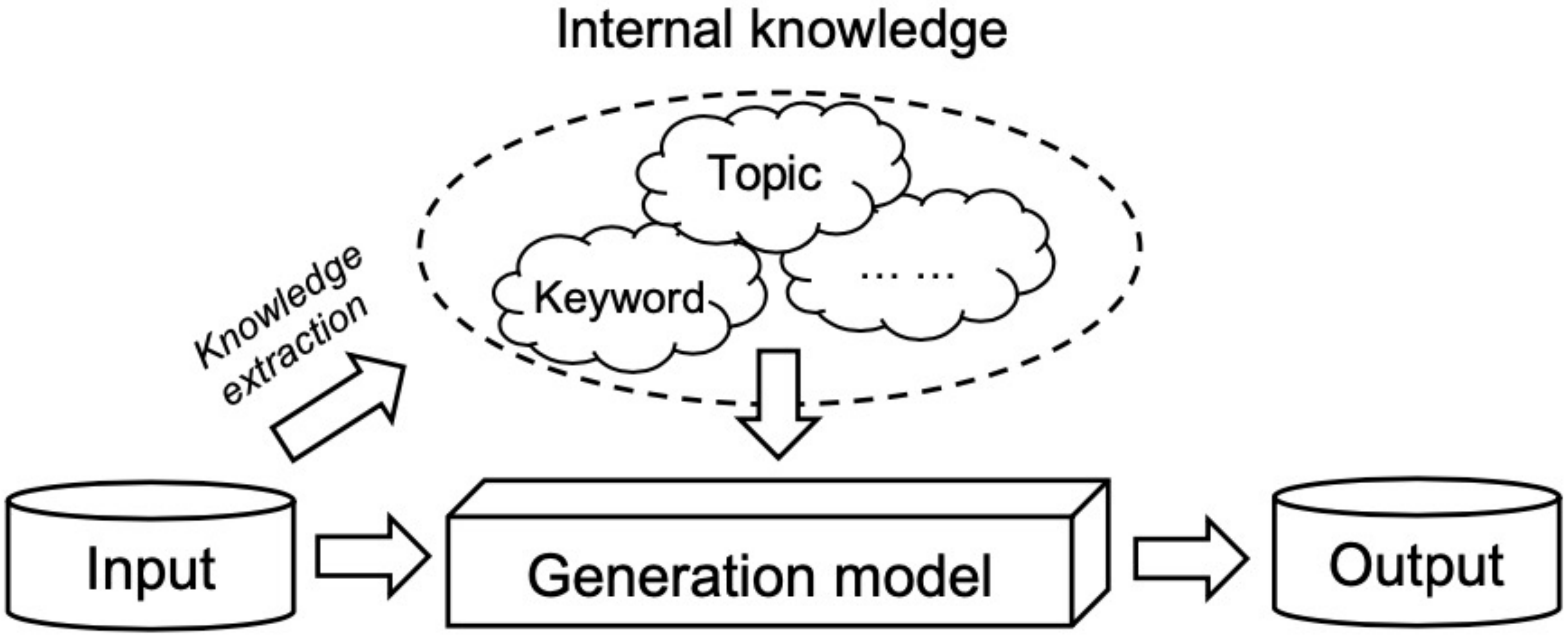
Source: Jay Alammar (2019), The Illustrated GPT-2 (Visualizing Transformer Language Models), https://jalammar.github.io/illustrated-gpt2/

T5

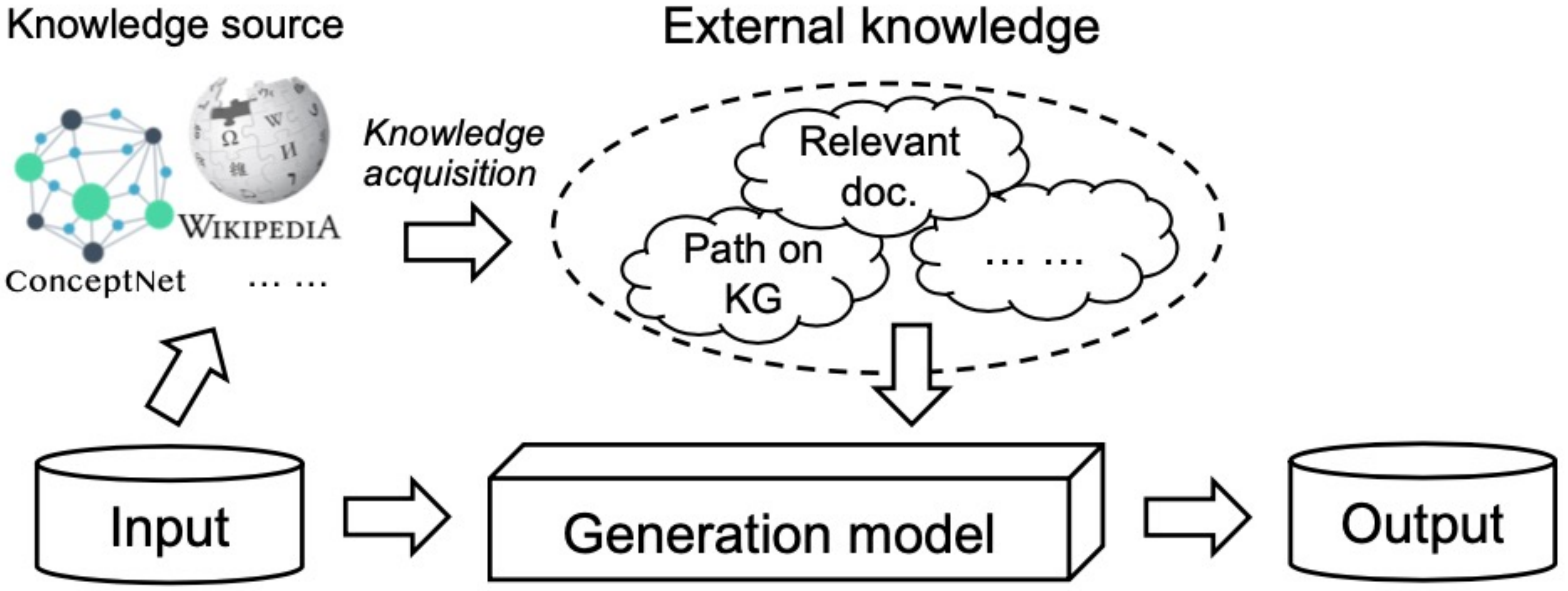
Text-to-Text Transfer Transformer


"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

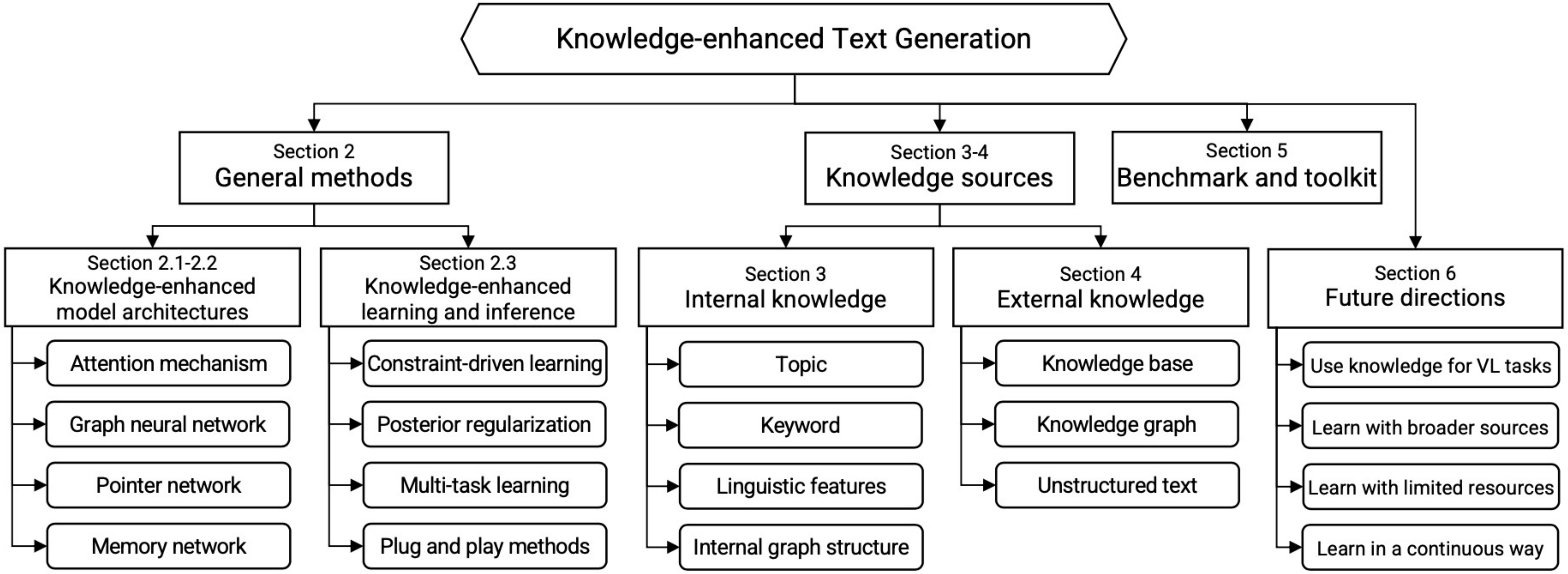
Internal Knowledge-Enhanced Text Generation



External Knowledge-Enhanced Text Generation



Knowledge-Enhanced Text Generation



Text Generation Models

- Encoder-decoder frameworks
  - Recurrent Neural Network (RNN)
    - RNN- Seq2Seq
  - Convolutional neural network (CNN) based encoder-decoder
  - Transformer encoder-decoder

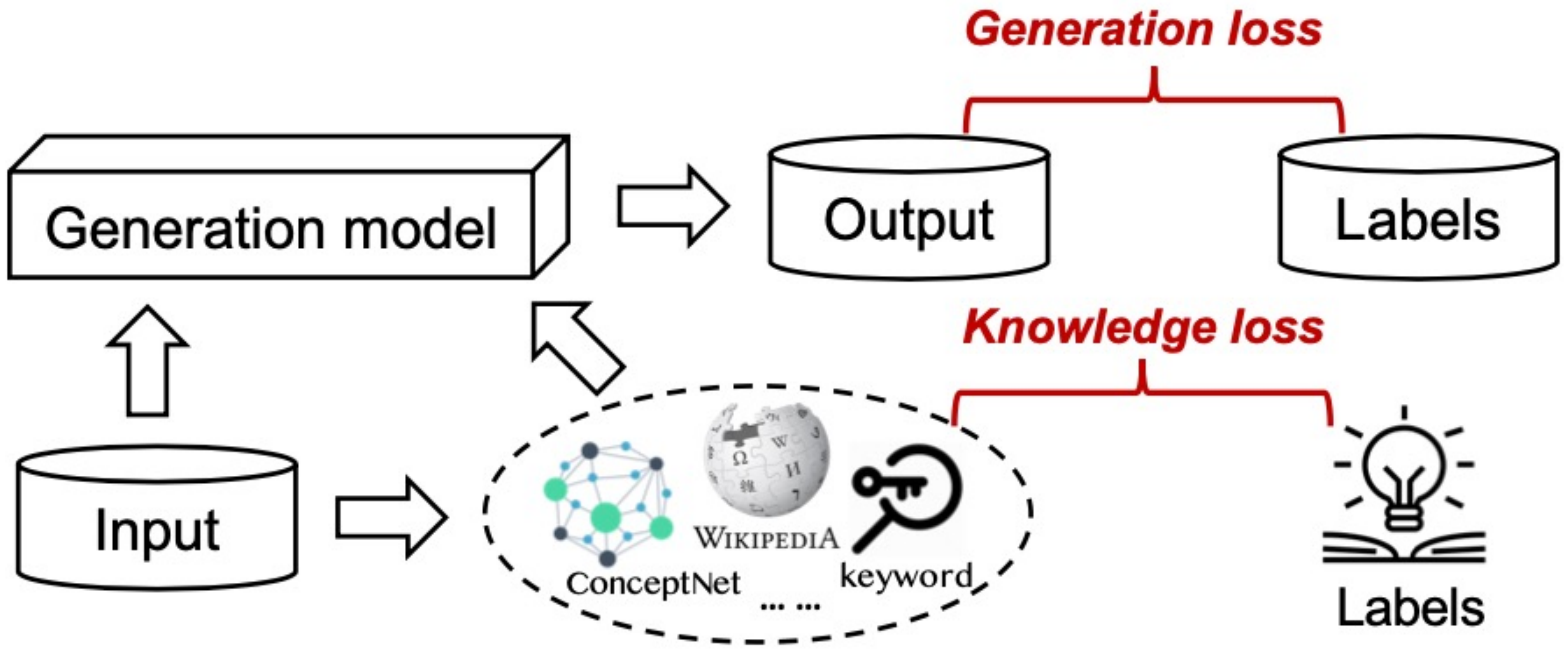
Source: Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang (2022). "A survey of knowledge-enhanced text generation." ACM Computing Surveys (2022).

Text Generation
Encoder-Decoder Frameworks
Conditional Distribution

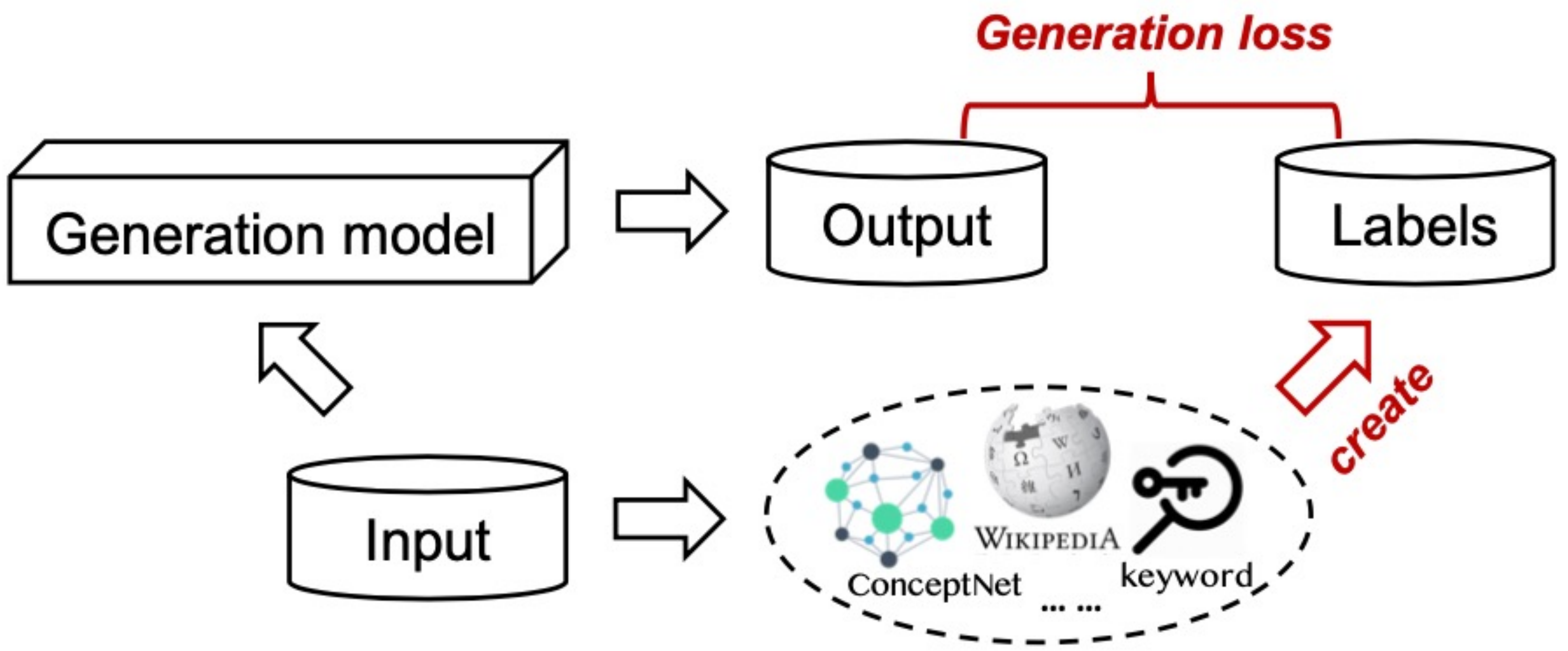
$$P(Y|X) = P(y_1, \cdots, y_m | x_1, \cdots, x_n) = \prod_{t=1}^{m} p(y_t | X, y_1, \cdots, y_{t-1})$$

Knowledge-Enhanced Text Generation

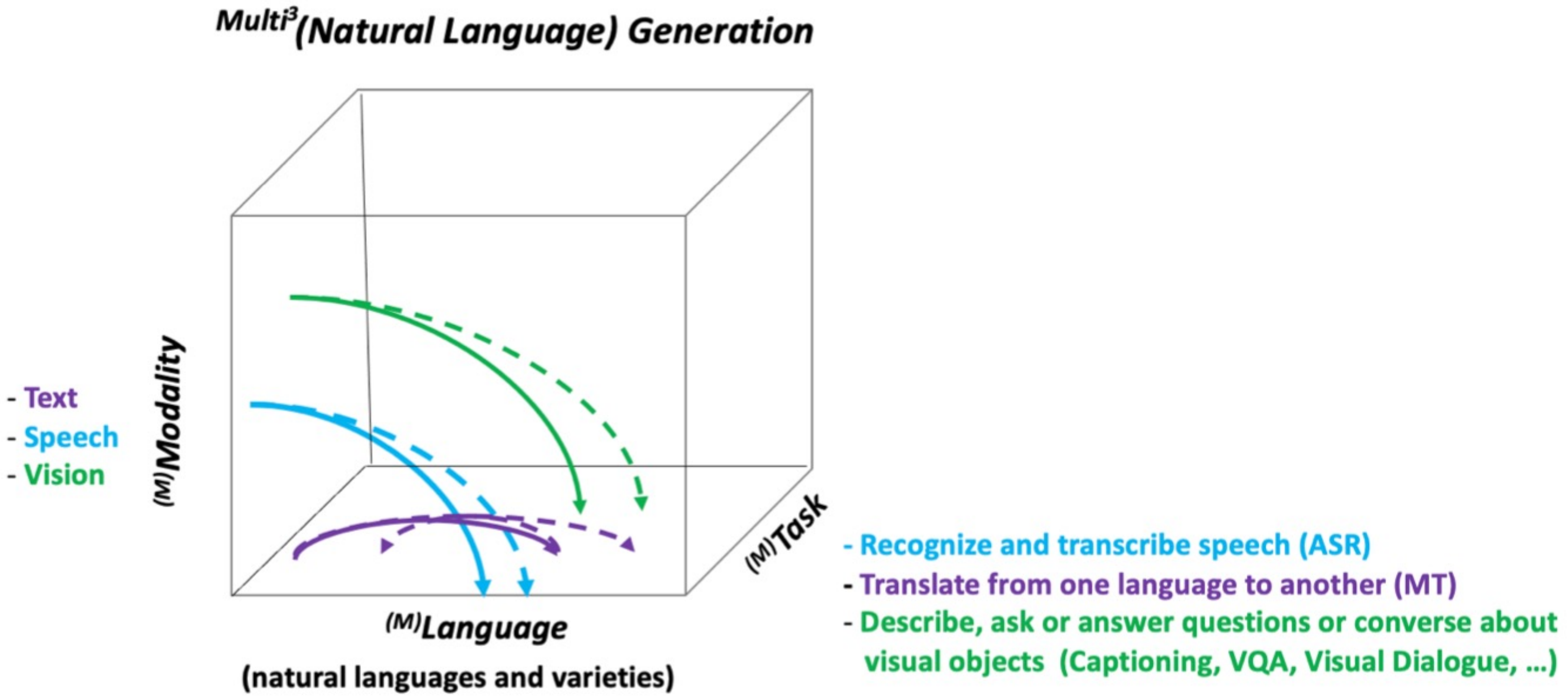
Incorporating knowledge into text generation by treating knowledge as the target



Knowledge-Enhanced Text Generation



Neural Natural Language Generation: Multilinguality, Multimodality, Controllability and Learning

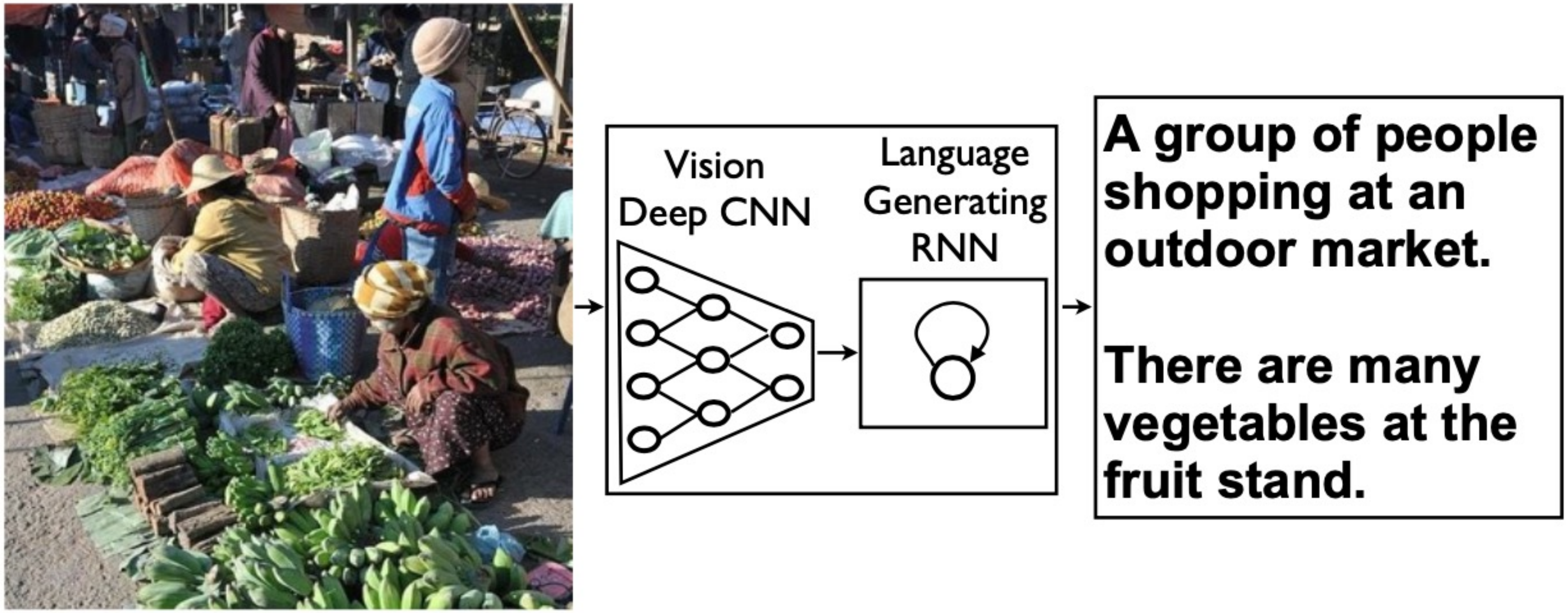

Multi³(Natural Language) Generation
- Text
- Speech
- Vision
(M)Modality
(M)Task
(M)Language
(natural languages and varieties)

- Recognize and transcribe speech (ASR)
- Translate from one language to another (MT)
- Describe, ask or answer questions or converse about visual objects (Captioning, VQA, Visual Dialogue, ...)

Source: Erkut Erdem, Menekse Kuyu, Semih Yagcioglu, Anette Frank, Letitia Parcalabescu, Barbara Plank, Andrii Babii et al (2022). "Neural Natural Language Generation: A Survey on Multilinguality, Multimodality, Controllability and Learning." Journal of Artificial Intelligence Research 73 (2022): 1131-1207.

Neural Image Captioning (NIC)

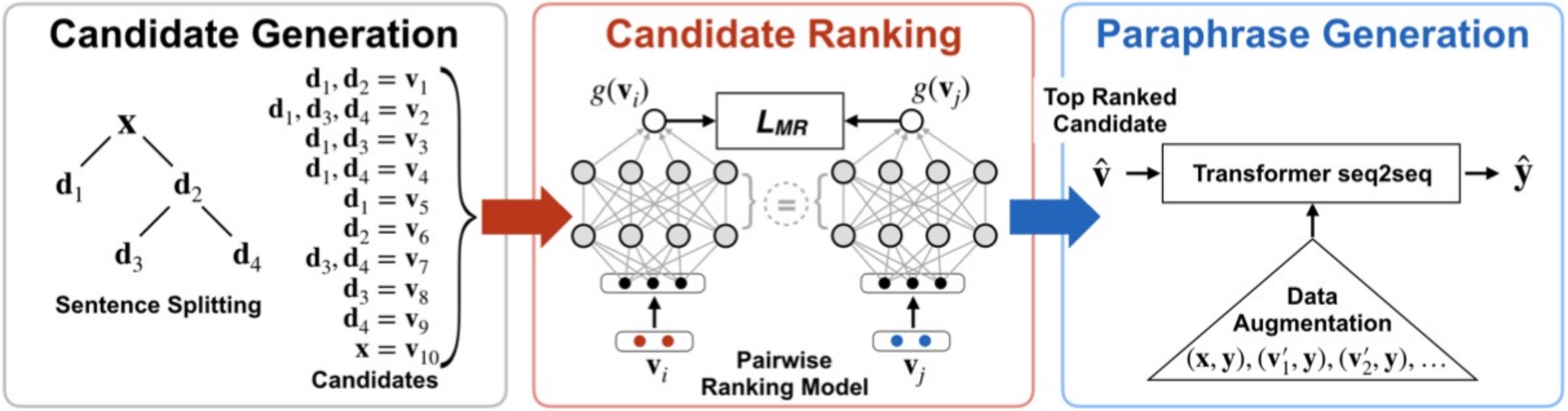
image-to-text description generation


Vision
Deep CNN
Language
Generating
RNN
A group of people shopping at an outdoor market.
There are many vegetables at the fruit stand.

Source: Erkut Erdem, Menekse Kuyu, Semih Yagcioglu, Anette Frank, Letitia Parcalabescu, Barbara Plank, Andrii Babii et al (2022). "Neural Natural Language Generation: A Survey on Multilinguality, Multimodality, Controllability and Learning." Journal of Artificial Intelligence Research 73 (2022): 1131-1207.

Controllable Text Simplification

with an explicit paraphrasing pipeline



INPUT $\mathbf{x}$: *The exhibition, which opened Oct. 8 and runs through Jan. 3, features 27 self-portraits.* **REFERENCE** $\mathbf{y}$: *The show started Oct. 8. It ends Jan. 3.*

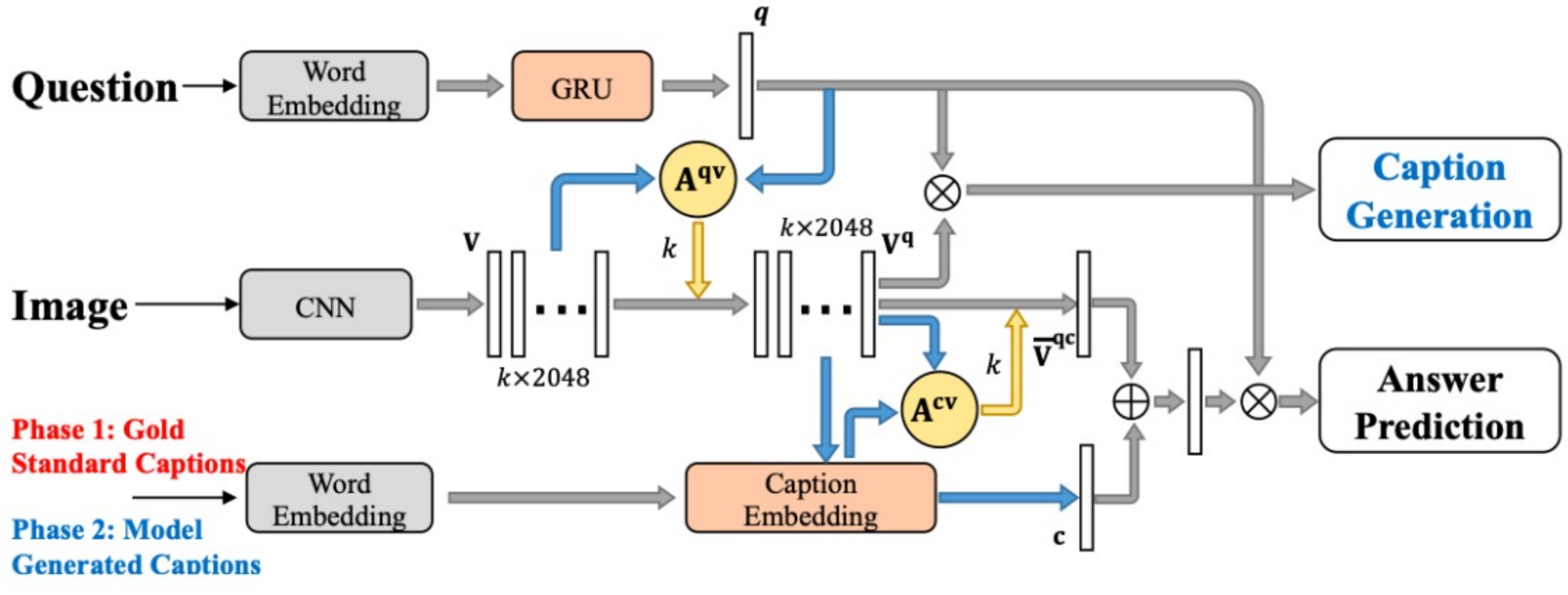
$\mathbf{d}_1$: *The exhibition features 27 self-portraits.* $\mathbf{d}_2$: *The exhibition opened Oct. 8 and runs through Jan. 3.*

$\mathbf{d}_3$: *The exhibition opened Oct. 8.* $\mathbf{d}_4$: *The exhibition runs through Jan. 3.* $\hat{\mathbf{v}} = \mathbf{v}_7$: *The exhibition opened Oct. 8. The exhibition runs through Jan. 3.*

Source: Erkut Erdem, Menekse Kuyu, Semih Yagcioglu, Anette Frank, Letitia Parcalabescu, Barbara Plank, Andrii Babii et al (2022). "Neural Natural Language Generation: A Survey on Multilinguality, Multimodality, Controllability and Learning." Journal of Artificial Intelligence Research 73 (2022): 1131-1207.

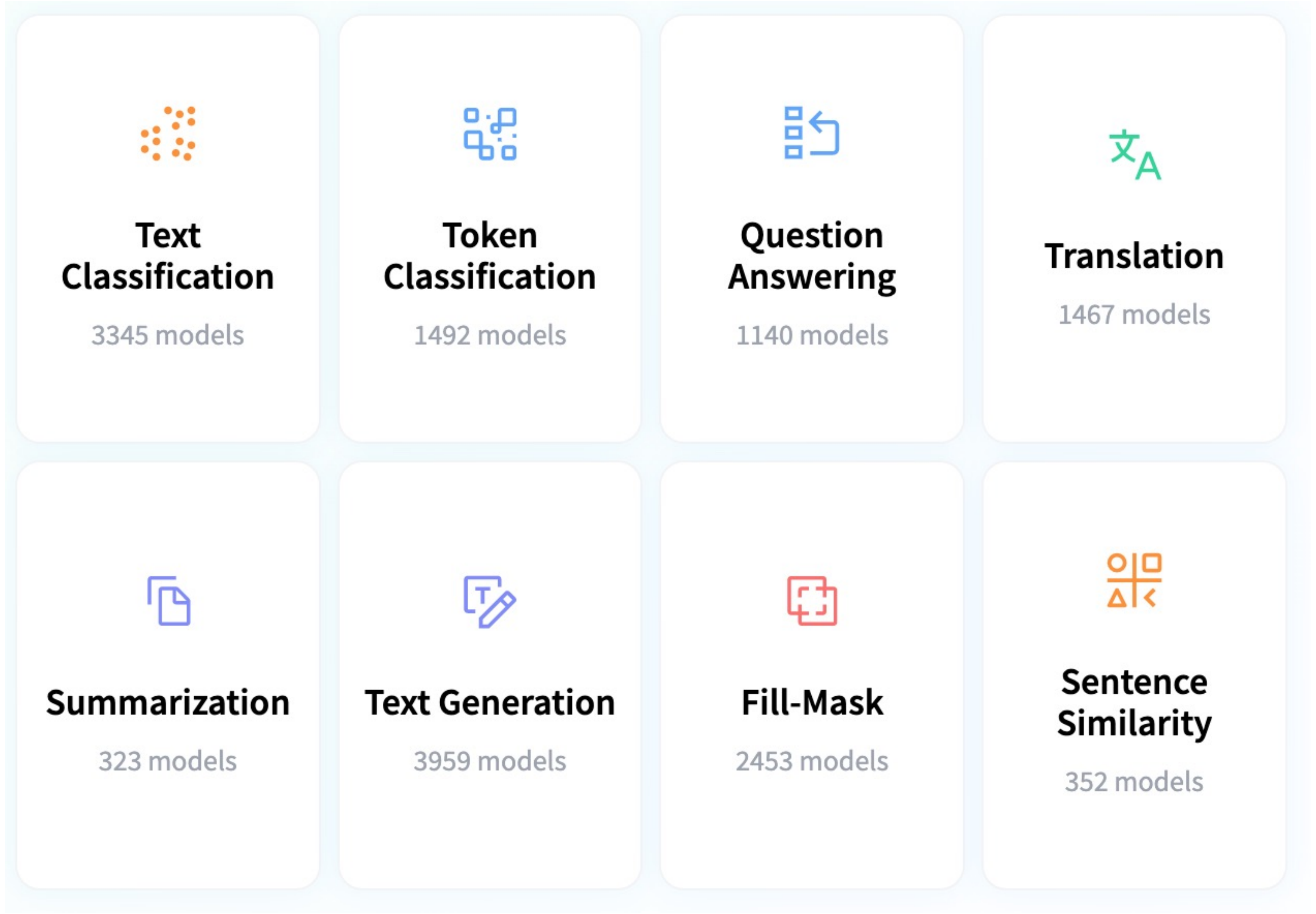
Visual Question Answering

Neural caption generation is employed to aid answer prediction



Source: Erkut Erdem, Menekse Kuyu, Semih Yagcioglu, Anette Frank, Letitia Parcalabescu, Barbara Plank, Andrii Babii et al (2022). "Neural Natural Language Generation: A Survey on Multilinguality, Multimodality, Controllability and Learning." Journal of Artificial Intelligence Research 73 (2022): 1131-1207.

Hugging Face Tasks
Natural Language Processing



Text Classification
3345 models

Token Classification
1492 models

Question Answering
1140 models

Translation
1467 models

Summarization
323 models

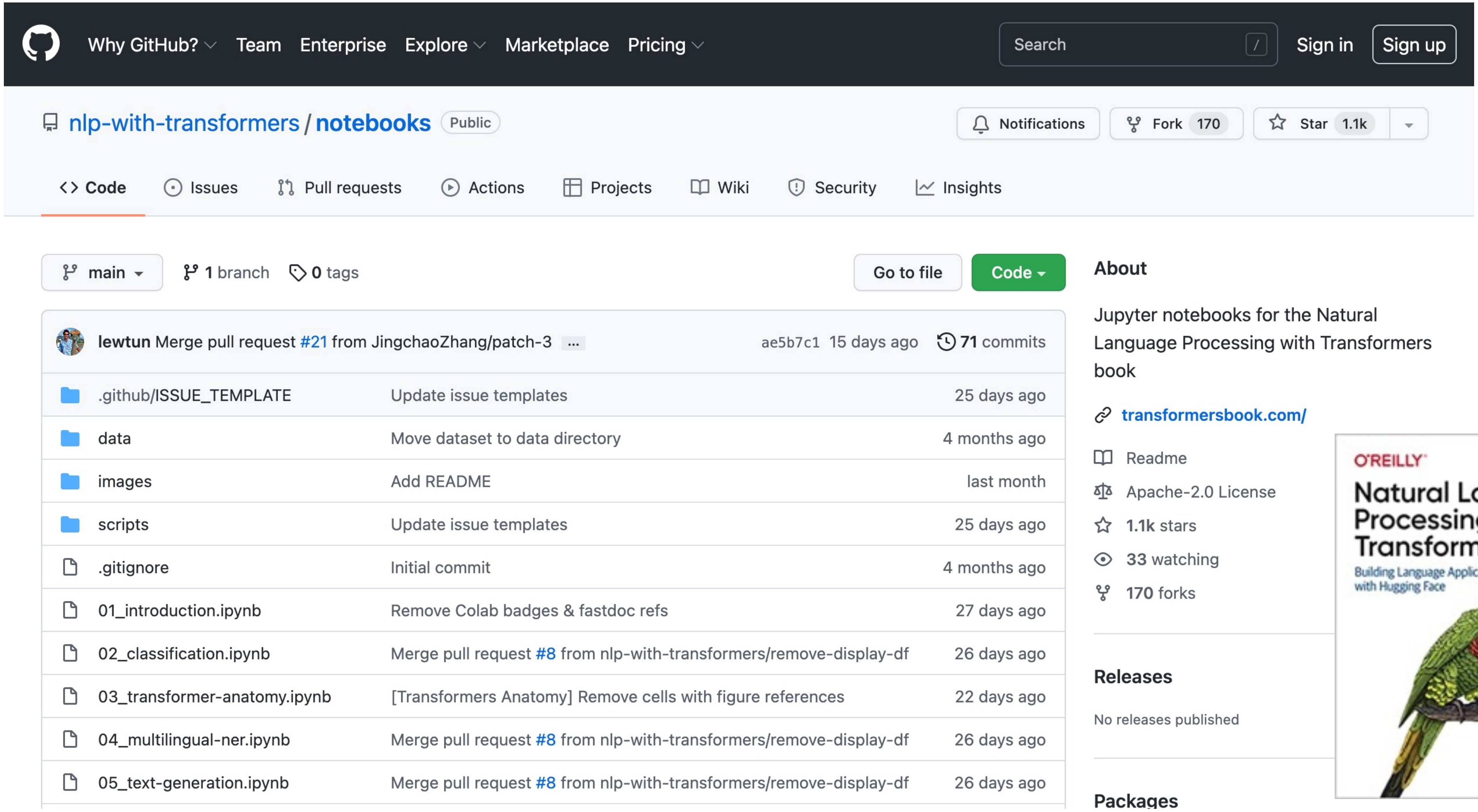
Text Generation
3959 models

Fill-Mask
2453 models

Sentence Similarity
352 models

https://huggingface.co/tasks

NLP with Transformers Github



nlp-with-transformers / notebooks Public

1 branch 0 tags

lewtun Merge pull request #21 from JingchaoZhang/patch-3 … ae5b7c1 15 days ago 71 commits

.github/ISSUE_TEMPLATE	Update issue templates	25 days ago
data	Move dataset to data directory	4 months ago
images	Add README	last month
scripts	Update issue templates	25 days ago
.gitignore	Initial commit	4 months ago
01_introduction.ipynb	Remove Colab badges & fastdoc refs	27 days ago
02_classification.ipynb	Merge pull request #8 from nlp-with-transformers/remove-display-df	26 days ago
03_transformer-anatomy.ipynb	[Transformers Anatomy] Remove cells with figure references	22 days ago
04_multilingual-ner.ipynb	Merge pull request #8 from nlp-with-transformers/remove-display-df	26 days ago
05_text-generation.ipynb	Merge pull request #8 from nlp-with-transformers/remove-display-df	26 days ago

About

Jupyter notebooks for the Natural Language Processing with Transformers book

transformersbook.com/

Readme

Apache-2.0 License

1.1k stars

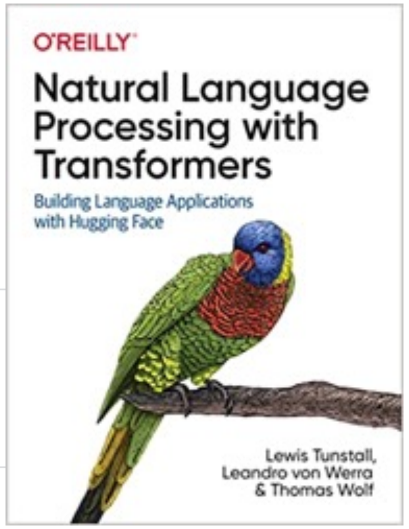
33 watching

170 forks

Releases

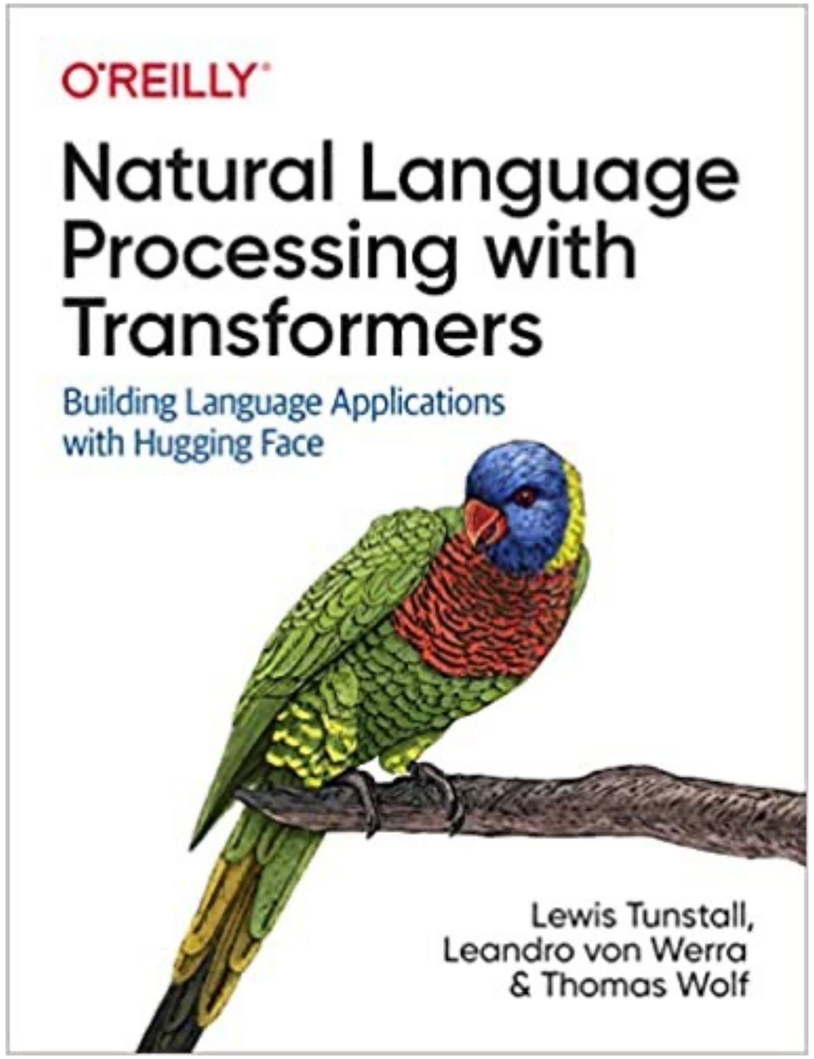
No releases published

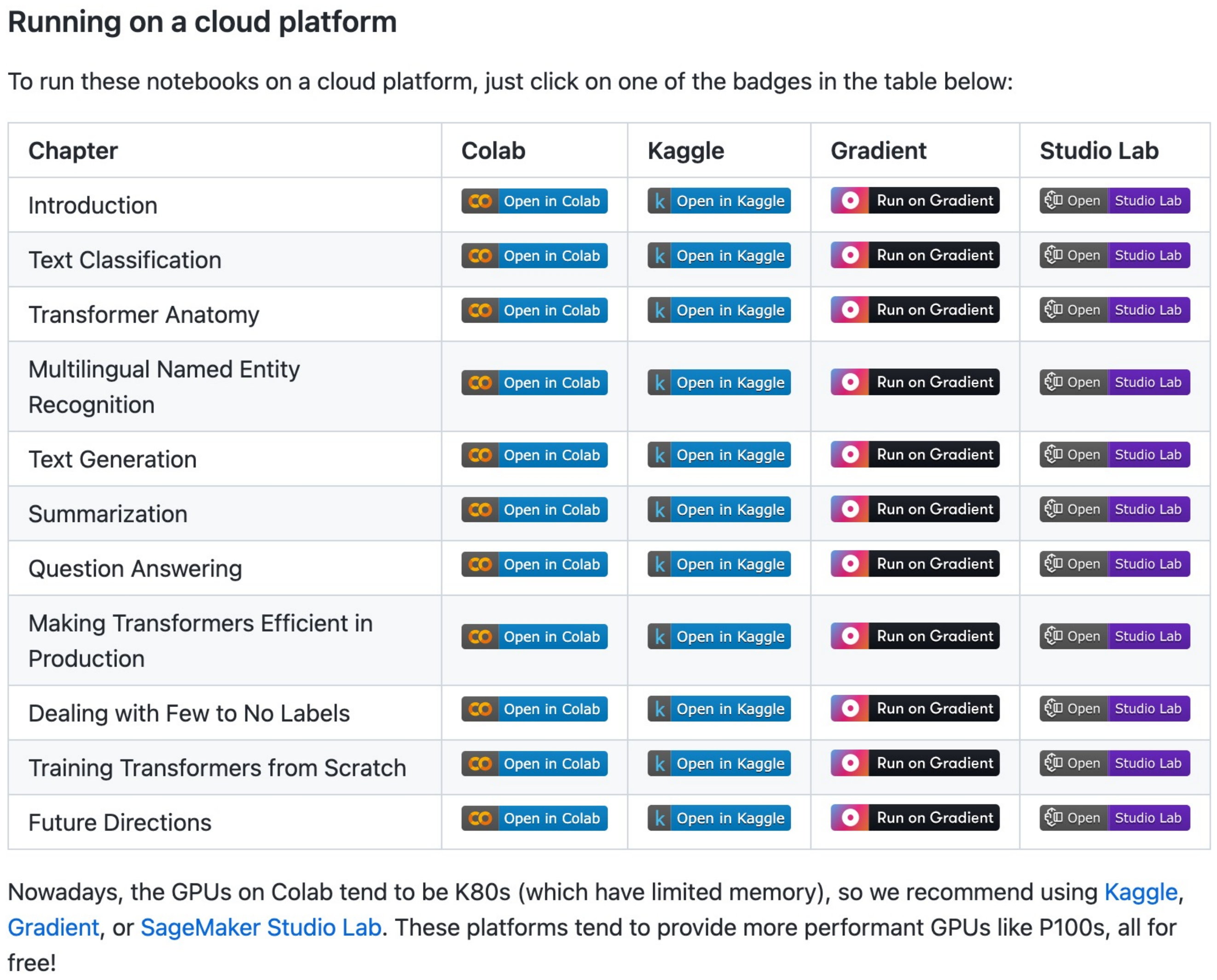
Packages


O'REILLY
Natural Language Processing with Transformers
Building Language Applications with Hugging Face
Lewis Tunstall, Leandro von Werra & Thomas Wolf

https://github.com/nlp-with-transformers/notebooks

NLP with Transformers Github Notebooks


O'REILLY
Natural Language Processing with Transformers
Building Language Applications with Hugging Face
Lewis Tunstall, Leandro von Werra & Thomas Wolf



Running on a cloud platform

To run these notebooks on a cloud platform, just click on one of the badges in the table below:

Chapter	Colab	Kaggle	Gradient	Studio Lab
Introduction	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Text Classification	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Transformer Anatomy	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Multilingual Named Entity Recognition	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Text Generation	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Summarization	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Question Answering	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Making Transformers Efficient in Production	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Dealing with Few to No Labels	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Training Transformers from Scratch	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Future Directions	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab

Nowadays, the GPUs on Colab tend to be K80s (which have limited memory), so we recommend using Kaggle, Gradient, or SageMaker Studio Lab. These platforms tend to provide more performant GPUs like P100s, all for free!

https://github.com/nlp-with-transformers/notebooks

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

NLP with Transformers

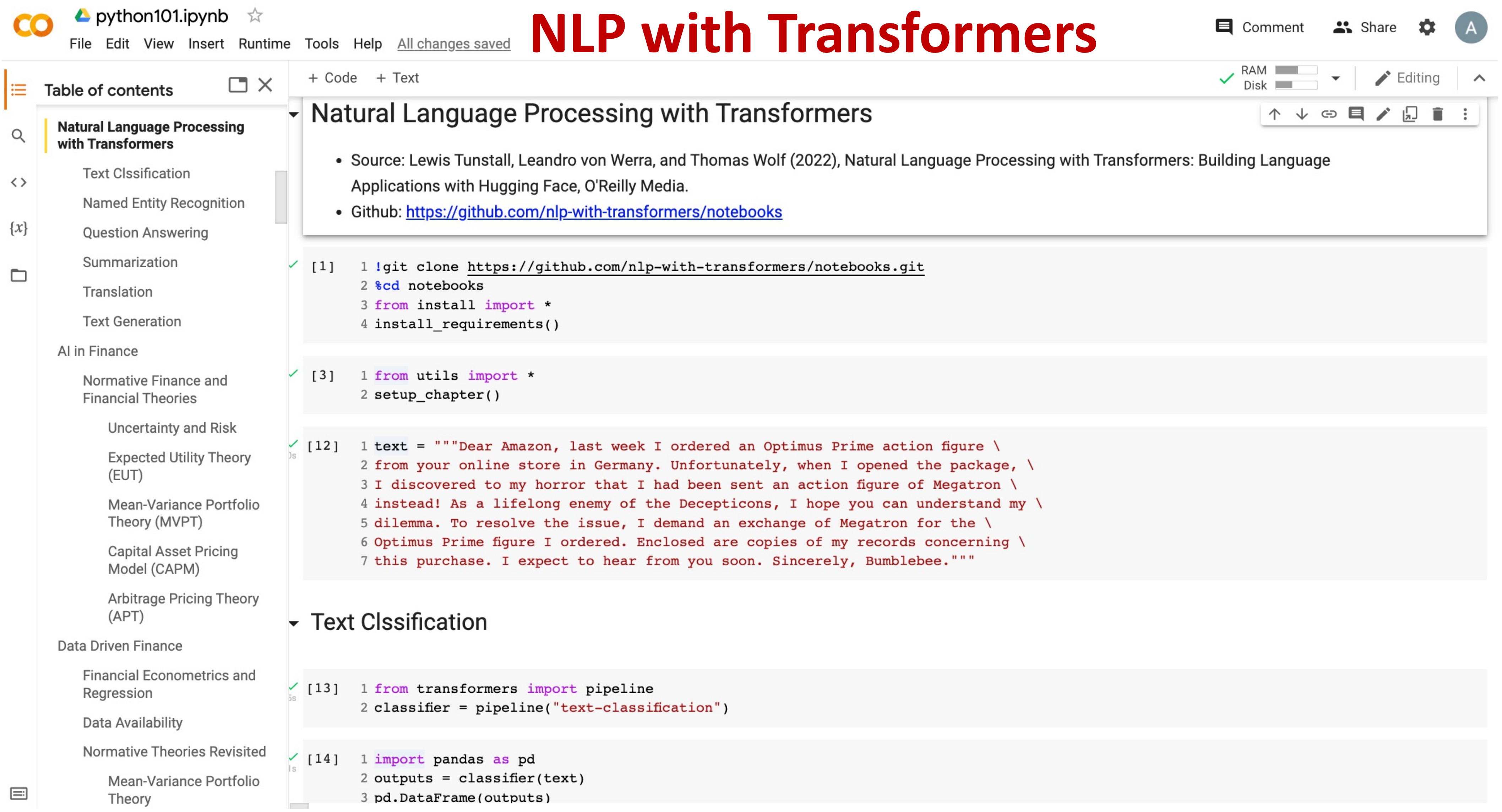


Table of contents

Natural Language Processing with Transformers

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: https://github.com/nlp-with-transformers/notebooks

[1]
```
!git clone https://github.com/nlp-with-transformers/notebooks.git
%cd notebooks
from install import *
install_requirements()
```

[3]
```
from utils import *
setup_chapter()
```

[12]
```
text = """Dear Amazon, last week I ordered an Optimus Prime action figure \
from your online store in Germany. Unfortunately, when I opened the package, \
I discovered to my horror that I had been sent an action figure of Megatron \
instead! As a lifelong enemy of the Decepticons, I hope you can understand my \
dilemma. To resolve the issue, I demand an exchange of Megatron for the \
Optimus Prime figure I ordered. Enclosed are copies of my records concerning \
this purchase. I expect to hear from you soon. Sincerely, Bumblebee."""
```

Text Clssification

[13]
```
from transformers import pipeline
classifier = pipeline("text-classification")
```

[14]
```
import pandas as pd
outputs = classifier(text)
pd.DataFrame(outputs)
```

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

Text Classification

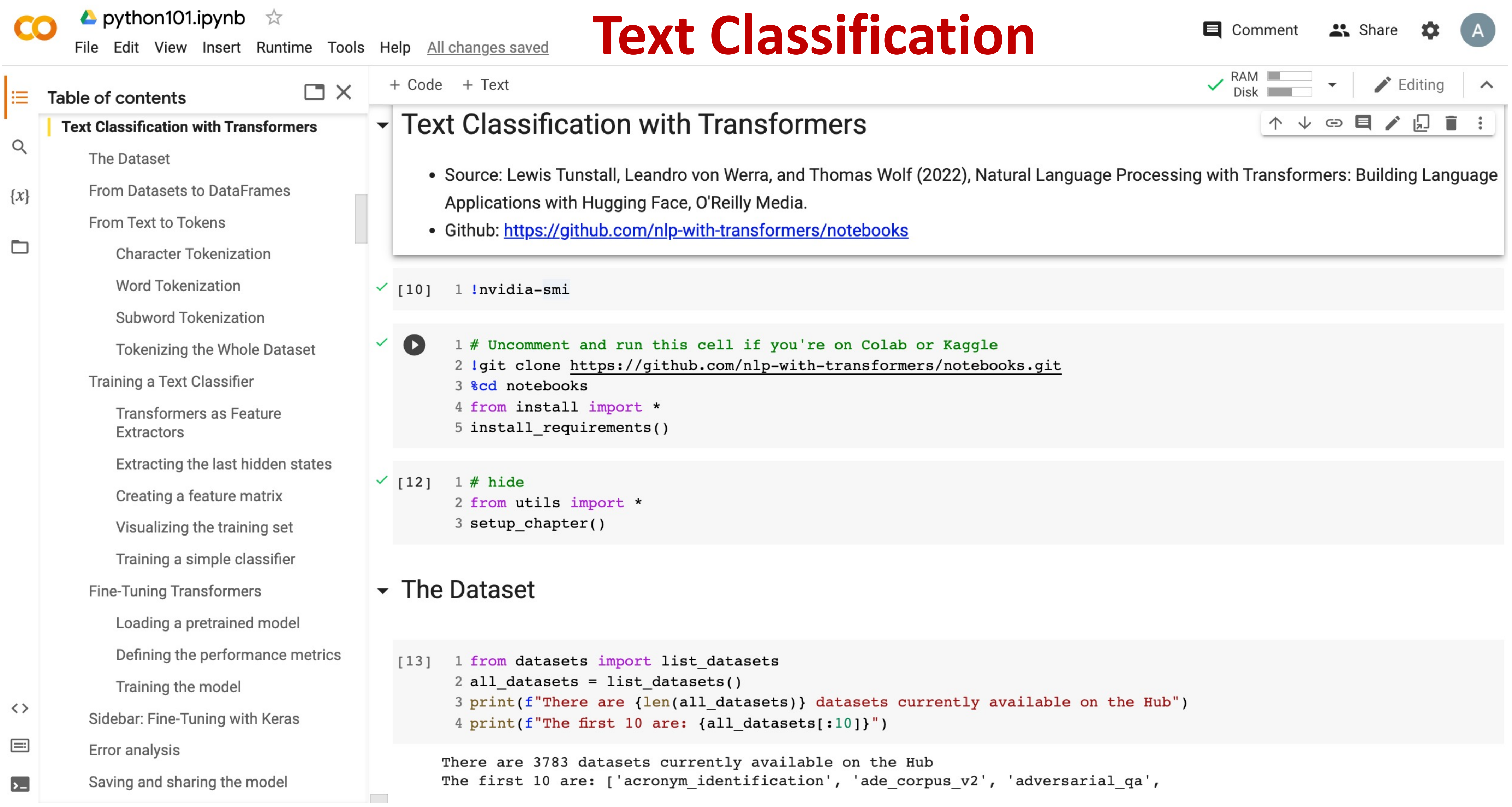


Table of contents

Text Classification with Transformers

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: https://github.com/nlp-with-transformers/notebooks

```
[10] !nvidia-smi
```

```
# Uncomment and run this cell if you're on Colab or Kaggle
!git clone https://github.com/nlp-with-transformers/notebooks.git
%cd notebooks
from install import *
install_requirements()
```

```
[12] # hide
from utils import *
setup_chapter()
```

The Dataset

```
[13] from datasets import list_datasets
all_datasets = list_datasets()
print(f"There are {len(all_datasets)} datasets currently available on the Hub")
print(f"The first 10 are: {all_datasets[:10]}")
```

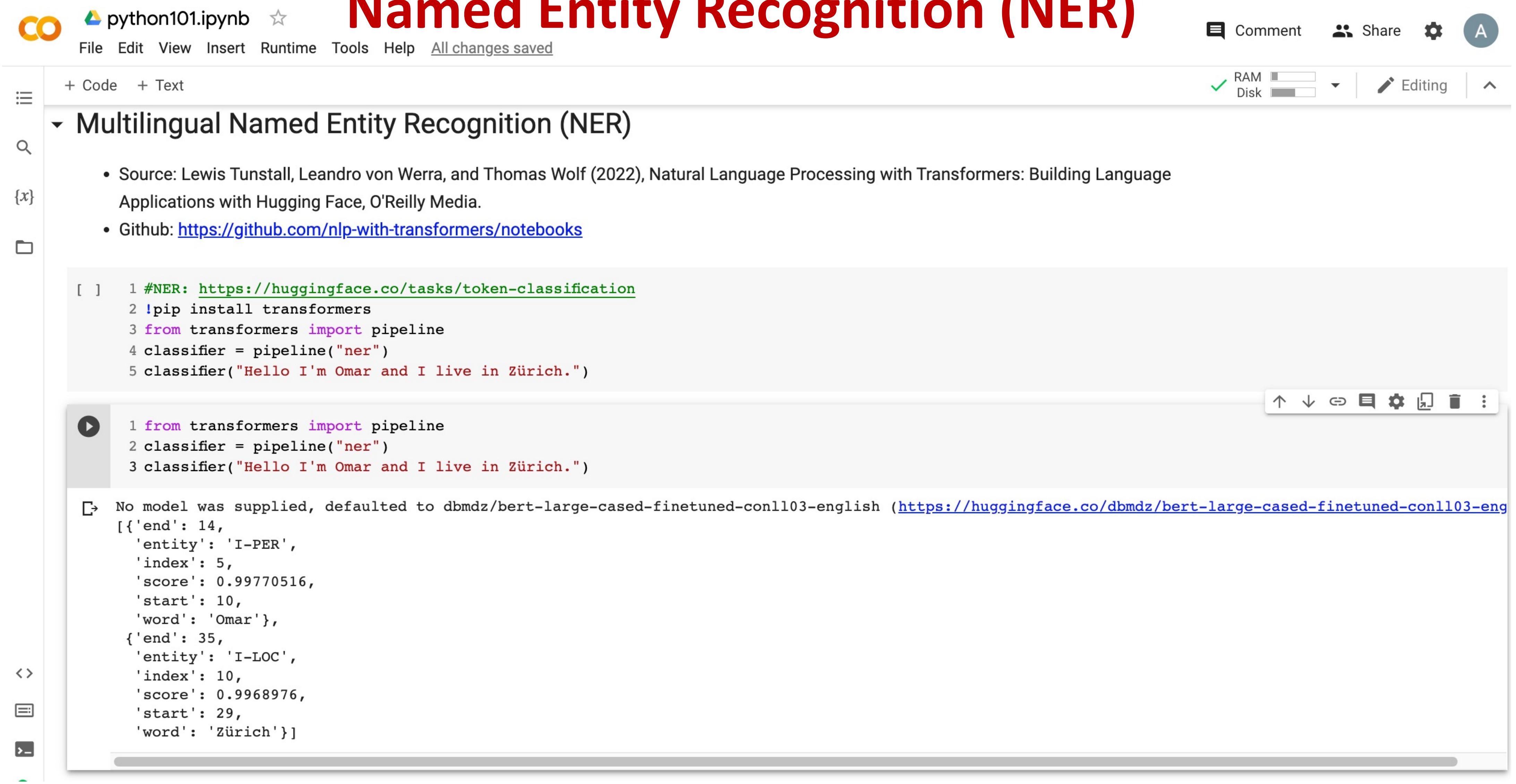
```
There are 3783 datasets currently available on the Hub
The first 10 are: ['acronym_identification', 'ade_corpus_v2', 'adversarial_qa',
```

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

Named Entity Recognition (NER)



Multilingual Named Entity Recognition (NER)

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: https://github.com/nlp-with-transformers/notebooks

```
#NER: https://huggingface.co/tasks/token-classification
!pip install transformers
from transformers import pipeline
classifier = pipeline("ner")
classifier("Hello I'm Omar and I live in Zürich.")
```

```
from transformers import pipeline
classifier = pipeline("ner")
classifier("Hello I'm Omar and I live in Zürich.")
```

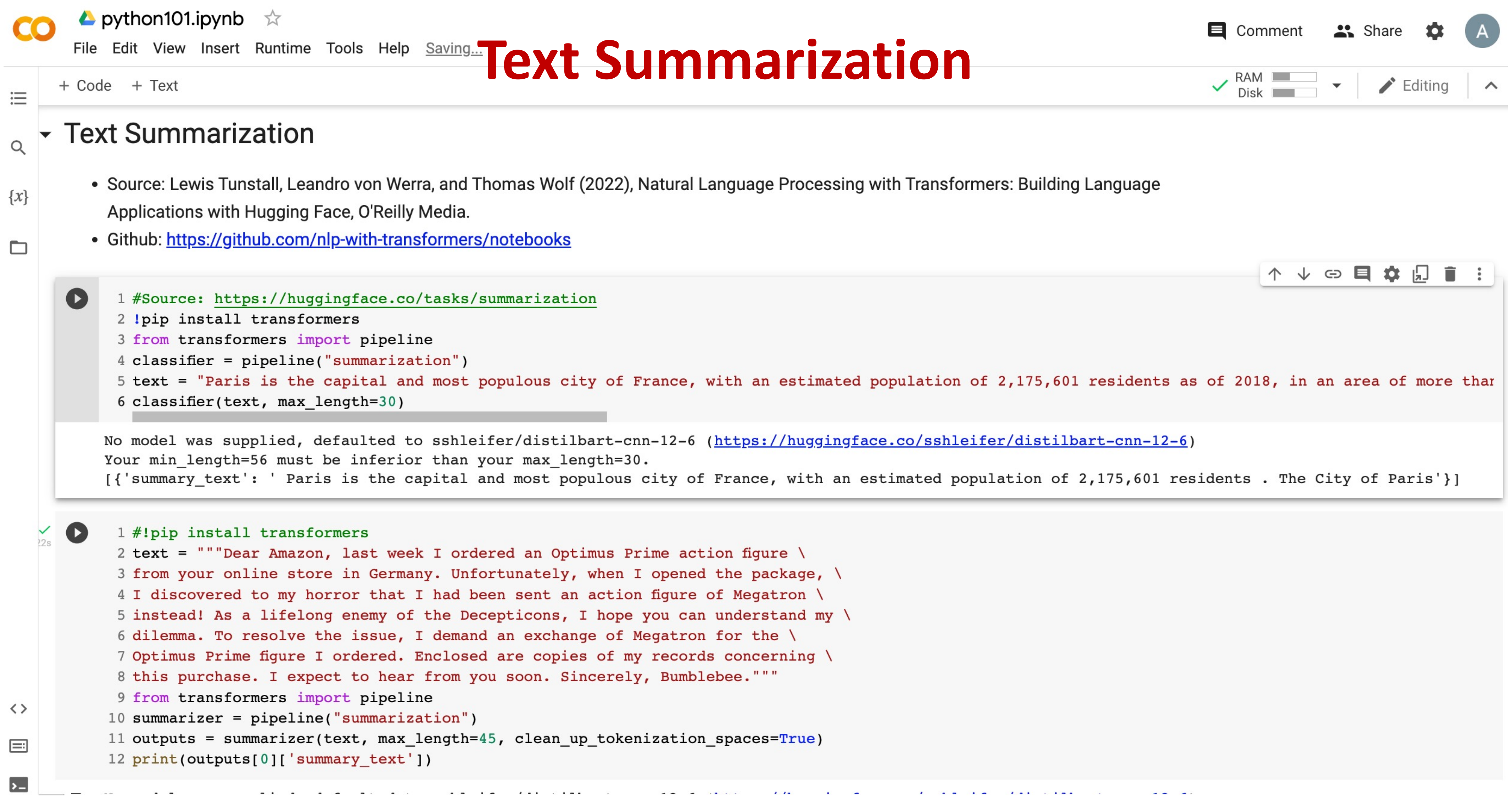
```
No model was supplied, defaulted to dbmdz/bert-large-cased-finetuned-conll03-english (https://huggingface.co/dbmdz/bert-large-cased-finetuned-conll03-eng
[{'end': 14,
  'entity': 'I-PER',
  'index': 5,
  'score': 0.99770516,
  'start': 10,
  'word': 'Omar'},
 {'end': 35,
  'entity': 'I-LOC',
  'index': 10,
  'score': 0.9968976,
  'start': 29,
  'word': 'Zürich'}]
```

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

Text Summarization



Text Summarization

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: https://github.com/nlp-with-transformers/notebooks

```
#Source: https://huggingface.co/tasks/summarization
!pip install transformers
from transformers import pipeline
classifier = pipeline("summarization")
text = "Paris is the capital and most populous city of France, with an estimated population of 2,175,601 residents as of 2018, in an area of more thar
classifier(text, max_length=30)
```

```
No model was supplied, defaulted to sshleifer/distilbart-cnn-12-6 (https://huggingface.co/sshleifer/distilbart-cnn-12-6)
Your min_length=56 must be inferior than your max_length=30.
[{'summary_text': ' Paris is the capital and most populous city of France, with an estimated population of 2,175,601 residents . The City of Paris'}]
```

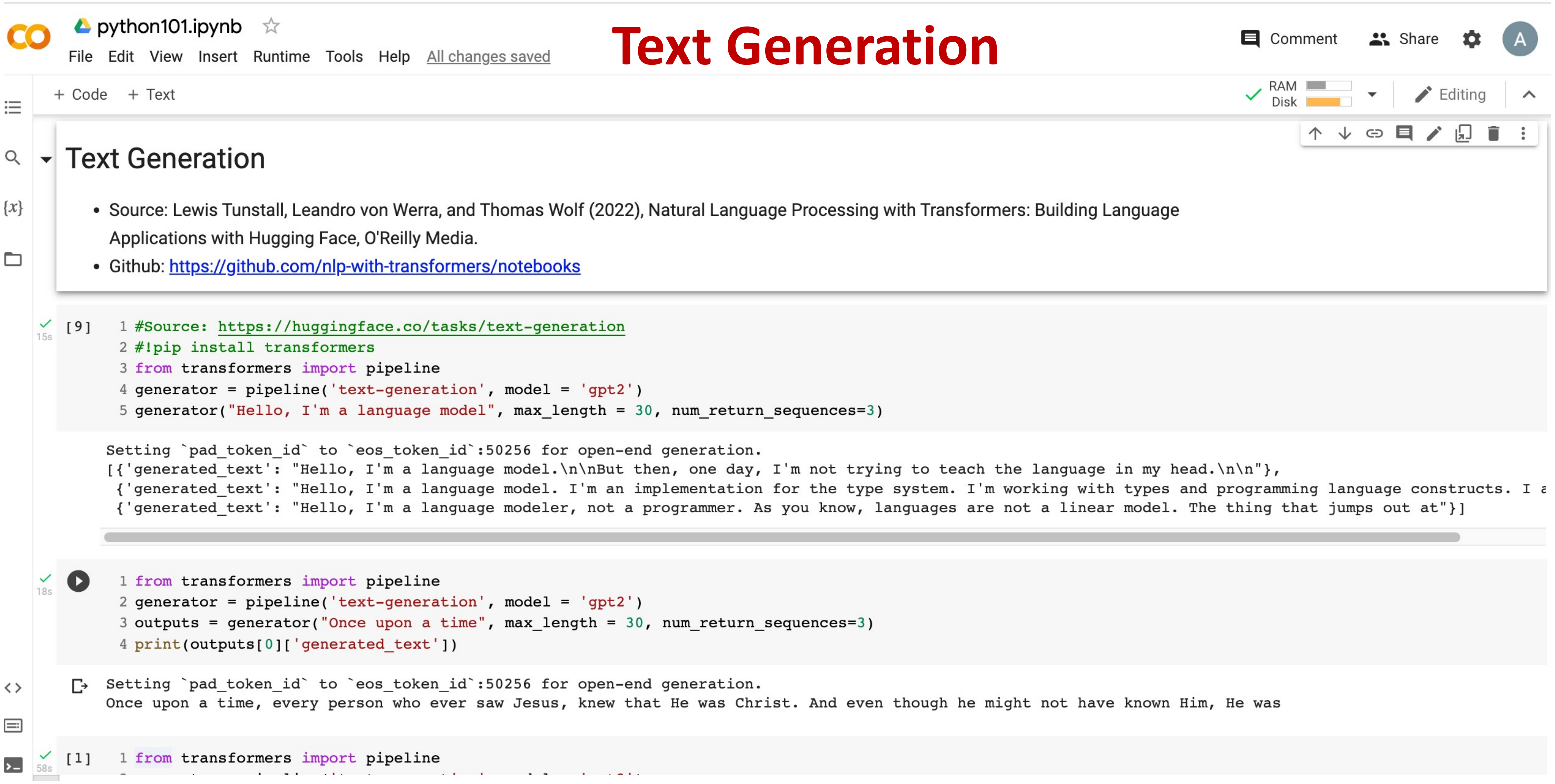
```
#!pip install transformers
text = """Dear Amazon, last week I ordered an Optimus Prime action figure \
from your online store in Germany. Unfortunately, when I opened the package, \
I discovered to my horror that I had been sent an action figure of Megatron \
instead! As a lifelong enemy of the Decepticons, I hope you can understand my \
dilemma. To resolve the issue, I demand an exchange of Megatron for the \
Optimus Prime figure I ordered. Enclosed are copies of my records concerning \
this purchase. I expect to hear from you soon. Sincerely, Bumblebee."""
from transformers import pipeline
summarizer = pipeline("summarization")
outputs = summarizer(text, max_length=45, clean_up_tokenization_spaces=True)
print(outputs[0]['summary_text'])
```

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

Text Generation



Text Generation

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: https://github.com/nlp-with-transformers/notebooks

```
[9]
#Source: https://huggingface.co/tasks/text-generation
#!pip install transformers
from transformers import pipeline
generator = pipeline('text-generation', model = 'gpt2')
generator("Hello, I'm a language model", max_length = 30, num_return_sequences=3)
```

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
[{'generated_text': "Hello, I'm a language model.\n\nBut then, one day, I'm not trying to teach the language in my head.\n\n"},
 {'generated_text': "Hello, I'm a language model. I'm an implementation for the type system. I'm working with types and programming language constructs. I a
 {'generated_text': "Hello, I'm a language modeler, not a programmer. As you know, languages are not a linear model. The thing that jumps out at"}]
```

```
from transformers import pipeline
generator = pipeline('text-generation', model = 'gpt2')
outputs = generator("Once upon a time", max_length = 30, num_return_sequences=3)
print(outputs[0]['generated_text'])
```

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
Once upon a time, every person who ever saw Jesus, knew that He was Christ. And even though he might not have known Him, He was
```

```
[1]
from transformers import pipeline
```

https://tinyurl.com/aintpupython101

Text Generation

```
!pip install transformers
from transformers import pipeline
generator = pipeline('text-generation', model = 'gpt2')
generator("Hello, I'm a language model", max_length = 30, num_return_sequences=3)
```

```
[{'generated_text': "Hello, I'm a language model. It's like looking at it,
where is each word of the sentence? That's what I mean. Like"},
{'generated_text': "Hello, I'm a language modeler. I'm using this for two
purposes: I'm having a lot fewer bugs and faster performance. If I"},
{'generated_text': 'Hello, I\'m a language model, and I was born to
code."\n\nNow, I am thinking about this from a different perspective with
a'}]
```

https://tinyurl.com/aintpupython101

Text Generation

```python
from transformers import pipeline
generator = pipeline('text-generation', model = 'gpt2')
outputs = generator("Once upon a time", max_length = 30)
print(outputs[0]['generated_text'])
```

```
Once upon a time, every person who ever saw
Jesus, knew that He was Christ. And even
though he might not have known Him, He was
```

Text Generation

```python
from transformers import pipeline
generator = pipeline('text-generation', model = 'gpt2')
outputs = generator("Once upon a time", max_length = 100)
print(outputs[0]['generated_text'])
```

```
Once upon a time we should be able to speak to people who
have lost children, so we try to take those that have lost
the children to our institutions — but the first time is very
hard for us because of our institutions. To me, it's
important to acknowledge that in an institution of faith and
love they are not children. And that there are many people
who are still hurting the child and there are many in need of
help, if not a system. So I'm very curious
```

https://tinyurl.com/aintpupython101

Text2Text Generation

```python
from transformers import pipeline
text2text_generator = pipeline("text2text-generation", model = 't5-base')
outputs = text2text_generator("translate from English to French: I am a student")
print(outputs[0]['generated_text'])
```

I am a student

Je suis un étudiant

https://tinyurl.com/aintpupython101

Text2Text Generation

```
from transformers import pipeline
text2text_generator = pipeline("text2text-generation")
text2text_generator("question: What is 42 ? context: 42 is the answer to life, the universe and everything")
```

```
[{'generated_text': 'the answer to life, the universe and everything'}]
```

https://tinyurl.com/aintpupython101

Summary

- **Text Generation**
  - **Natural Language Generation (NLG)**
    - Language Modeling
    - Conditional Language Modeling
  - **Next Word Prediction**
- **Decoding Algorithm**
  - **Greedy Search Decoding**
  - **Beam Search Decoding**
  - **Sampling Methods**
  - **Top-k and Nucleus Sampling**

References

- Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Denis Rothman (2021), Transformers for Natural Language Processing: Build innovative deep neural network architectures for NLP with Python, PyTorch, TensorFlow, BERT, RoBERTa, and more, Packt Publishing.
- Savaş Yıldırım and Meysam Asgari-Chenaghlu (2021), Mastering Transformers: Build state-of-the-art models from scratch with advanced natural language processing techniques, Packt Publishing.
- Sowmya Vajjala, Bodhisattwa Majumder, Anuj Gupta (2020), Practical Natural Language Processing: A Comprehensive Guide to Building Real-World NLP Systems, O'Reilly Media.
- Dipanjan Sarkar (2019), Text Analytics with Python: A Practitioner's Guide to Natural Language Processing, Second Edition. APress.
- Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang (2022). "A survey of knowledge-enhanced text generation." ACM Computing Surveys (2022).
- Erkut Erdem, Menekse Kuyu, Semih Yagcioglu, Anette Frank, Letitia Parcalabescu, Barbara Plank, Andrii Babii et al (2022). "Neural Natural Language Generation: A Survey on Multilinguality, Multimodality, Controllability and Learning." Journal of Artificial Intelligence Research 73 (2022): 1131-1207.
- Jay Alammar (2019), The Illustrated GPT-2 (Visualizing Transformer Language Models), https://jalammar.github.io/illustrated-gpt2/
- Benjamin Bengfort, Rebecca Bilbro, and Tony Ojeda (2018), Applied Text Analysis with Python: Enabling Language-Aware Data Products with Machine Learning, O'Reilly.
- Charu C. Aggarwal (2018), Machine Learning for Text, Springer.
- Gabe Ignatow and Rada F. Mihalcea (2017), An Introduction to Text Mining: Research Design, Data Collection, and Analysis, SAGE Publications.
- Rajesh Arumugam (2018), Hands-On Natural Language Processing with Python: A practical guide to applying deep learning architectures to your NLP applications, Packt.
- Jake VanderPlas (2016), Python Data Science Handbook: Essential Tools for Working with Data, O'Reilly Media.
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." arXiv preprint arXiv:1810.04805.
- The Super Duper NLP Repo, https://notebooks.quantumstat.com/
- Jay Alammar (2018), The Illustrated Transformer, http://jalammar.github.io/illustrated-transformer/
- Jay Alammar (2019), A Visual Guide to Using BERT for the First Time, http://jalammar.github.io/a-visual-guide-to-using-bert-for-the-first-time/
- NLP with Transformer, https://github.com/nlp-with-transformers/notebooks
- Min-Yuh Day (2022), Python 101, https://tinyurl.com/aintpupython101