Artificial Intelligence for Text Analytics

Text Generation
Natural Language Generation (NLG)

Min-Yuh Day, Ph.D,
Associate Professor

Institute of Information Management, National Taipei University

https://web.ntpu.edu.tw/~myday

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

https://meet.google.com/paj-zhhj-myv
**Syllabus**

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Subject/Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2022/02/22</td>
<td>Introduction to Artificial Intelligence for Text Analytics</td>
</tr>
<tr>
<td>2</td>
<td>2022/03/01</td>
<td>Foundations of Text Analytics: Natural Language Processing (NLP)</td>
</tr>
<tr>
<td>3</td>
<td>2022/03/08</td>
<td>Python for Natural Language Processing</td>
</tr>
<tr>
<td>4</td>
<td>2022/03/15</td>
<td>Natural Language Processing with Transformers</td>
</tr>
<tr>
<td>5</td>
<td>2022/03/22</td>
<td>Case Study on Artificial Intelligence for Text Analytics I</td>
</tr>
<tr>
<td>6</td>
<td>2022/03/29</td>
<td>Text Classification and Sentiment Analysis</td>
</tr>
<tr>
<td>Week</td>
<td>Date</td>
<td>Subject/Topics</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>7</td>
<td>2022/04/05</td>
<td>Tomb-Sweeping Day (Holiday, No Classes)</td>
</tr>
<tr>
<td>8</td>
<td>2022/04/12</td>
<td>Midterm Project Report</td>
</tr>
<tr>
<td>9</td>
<td>2022/04/19</td>
<td>Multilingual Named Entity Recognition (NER), Text Similarity and Clustering</td>
</tr>
<tr>
<td>10</td>
<td>2022/04/26</td>
<td>Text Summarization and Topic Models</td>
</tr>
<tr>
<td>11</td>
<td>2022/05/03</td>
<td>Text Generation</td>
</tr>
<tr>
<td>12</td>
<td>2022/05/10</td>
<td>Case Study on Artificial Intelligence for Text Analytics II</td>
</tr>
</tbody>
</table>
## Syllabus

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

💡 Text Generation demo

using gpt2

🔗 Text Generation

Once upon a time,

Compute

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

Step 1: Transformers are the most
Step 2: Transformers are the most popular
Step 3: Transformers are the most popular toys

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

Transformer

Encoder \rightarrow Decoder

DistilBERT \rightarrow BERT \rightarrow RoBERTa \rightarrow XLM \rightarrow ALBERT \rightarrow ELECTRA \rightarrow DeBERTa

XLM-R \rightarrow XLM \rightarrow ALBERT \rightarrow ELECTRA

Transformer

T5 \rightarrow BART \rightarrow M2M-100 \rightarrow BigBird

Transformer

GPT \rightarrow GPT-2 \rightarrow GPT-3 \rightarrow GPT-Neo \rightarrow GPT-J

CTRL

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

Learning via SGD during unsupervised pre-training

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

M: Which animal do you like?
I like cows.

M: I heard they go to college.
Cows go to college?

M: I heard that a cow went to Harvard.
What did the cow study?

M: Bovine sciences.

M: Do horses ever go to Harvard?
Text Generation
Decoding Algorithm

• Greedy Search Decoding
• Beam Search Decoding
• Sampling Methods
• Top-k and Nucleus Sampling
## Text Generation

**Greedy Search Decoding**

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

Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
The Illustrated GPT-2
(Visualizing Transformer Language Models)
Jay Alammar (2019)

GPT-2 Output

GPT-2 Autoregression

Transformer Encoder Decoder

GPT-2 Sizes

GPT-2 Sizes Hyperparameters

Transformer Encoder

THE TRANSFORMER

ENCODER BLOCK

Feed Forward Neural Network

Self-Attention

<table>
<thead>
<tr>
<th>robot</th>
<th>must</th>
<th>obey</th>
<th>orders</th>
<th>&lt;eos&gt;</th>
<th>&lt;pad&gt;</th>
<th>...</th>
<th>&lt;pad&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
<td>512</td>
</tr>
</tbody>
</table>

Transformer Decoder

DECODER BLOCK #2

Feed Forward Neural Network

Encoder-Decoder Self-Attention

Masked Self-Attention

Input

orders

Self-Attention and Masked-Self-Attention

Transformer Decoder

GPT-2 Layers

GPT-2 Simple Output

GPT-2 Simple Output

T5
Text-to-Text Transfer Transformer

"translate English to German: That is good."
"cola sentence: The course is jumping well."
"sts1 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..."
"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 source

Knowledge acquisition

External knowledge

Input → Generation model → Output

Text Generation Models

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

Text Generation
Encoder-Decoder Frameworks
Conditional Distribution

\[ P(Y|X) = \prod_{t=1}^{m} p(y_t|X, y_1, \ldots, y_{t-1}) \]

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

Neural Image Captioning (NIC)
image-to-text description generation

A group of people shopping at an outdoor market.
There are many vegetables at the fruit stand.

Controllable Text Simplification with an explicit paraphrasing pipeline

Visual Question Answering

Neural caption generation is employed to aid answer prediction

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

https://github.com/nlp-with-transformers/notebooks
NLP with Transformers Github Notebooks

Running on a cloud platform

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

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Colab</th>
<th>Kaggle</th>
<th>Gradient</th>
<th>Studio Lab</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Text Classification</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Transformer Anatomy</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Multilingual Named Entity Recognition</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Text Generation</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Summarization</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Question Answering</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Making Transformers Efficient in Production</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Dealing with Few to No Labels</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Training Transformers from Scratch</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
<tr>
<td>Future Directions</td>
<td>🚀 Open in Colab</td>
<td>🚀 Open in Kaggle</td>
<td>🚀 Run on Gradient</td>
<td>🚀 Open Studio Lab</td>
</tr>
</tbody>
</table>

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/1FEG6DnGvwfUbeo4z1zTunjMqf2RkCrT

NLP with Transformers

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

```
  cd notebooks
  from install import *
  install_requirements()

[3] from util import *
  setup_chapter()

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

Text Classification

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

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

https://tinyurl.com/aintpupython101
Text Classification

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

The Dataset

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

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

https://tinyurl.com/aintpupython101
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.

```python
# NER: [https://huggingface.co/tasks/token-classification](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.")
```

No model was supplied, defaulted to dbmdz/bert-large-cased-finetuned-conll03-english ([https://huggingface.co/dbmdz/bert-large-cased-finetuned-conll03-english](https://huggingface.co/dbmdz/bert-large-cased-finetuned-conll03-english))

```json
["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"]
```
Python in Google Colab (Python101)

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

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

```python
# 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 than...
6 classifier(text, max_length=30)

No model was supplied, default 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
2 text = '''Dear Amazon, last week I ordered an Optimus Prime action figure \n3 from your online store in Germany. Unfortunately, when I opened the package, \n4 I discovered to my horror that I had been sent an action figure of Megatron \n5 instead! As a lifelong enemy of the Decepticons, I hope you can understand my \n6 dilemma. To resolve the issue, I demand an exchange of Megatron for the \n7 Optimus Prime figure I ordered. Enclosed are copies of my records concerning \n8 this purchase. I expect to hear from you soon. Sincerely, Bumblebee.'''
9 from transformers import pipeline
10 summarizer = pipeline("summarization")
11 outputs = summarizer(text, max_length=45, clean_up_tokenization_spaces=True)
12 print(outputs[0]["summary_text"])
```

https://tinyurl.com/aintpupython101
Python in Google Colab (Python101)

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

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

```python
# Source: https://huggingface.co/tasks/text-generation
1 #pip install transformers
2 from transformers import pipeline
3 generator = pipeline('text-generation', model = 'gpt2')
4 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.

```python
'generated_text': "Hello, I’m a language model.\nBut then, one day, I’m not trying to teach the language in my head.\nI’m implementing for the type system. I’m working with types and programming language constructs. I,\n'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")
```

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

Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.

```python
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
```

https://tinyurl.com/aintpuppython101
```python
!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)
```

```json
[{
'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.'}
]```

```text
Now, I am thinking about this from a different perspective with a'
```
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
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
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
Text2Text Generation

```python
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-Chenaghlou (2021), Mastering Transformers: Build state-of-the-art models from scratch with advanced natural language processing techniques, Packt Publishing.
- 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.