Text Summarization and Topic Models

Min-Yuh Day, Ph.D, Associate Professor

Institute of Information Management, National Taipei University

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

1102AITA07
MBA, IM, NTPU (M5026) (Spring 2022)
Tue 2, 3, 4 (9:10-12:00) (B8F40)

https://meet.google.com/paj-zhhj-my4
<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>
<tr>
<td>Week</td>
<td>Date</td>
<td>Subject/Topics</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<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 Summarization and Topic Models
Outline

• Text Summarization
  • Extractive Text Summarization
  • Abstractive Text Summarization
    • PEGASUS: Abstractive Summarization

• Topic Models
  • Topic Modeling
  • Latent Dirichlet Allocation (LDA)
  • BERTopic
Text Summarization
The tower is 324 metres (1,063 ft) tall, about the same height as an 81-storey building, and the tallest structure in Paris. Its base is square, measuring 125 metres (410 ft) on each side. It was the first structure to reach a height of 300 metres. Excluding transmitters, the Eiffel Tower is the second tallest free-standing structure in France after the Millau Viaduct.

https://huggingface.co/tasks/summarization
The tower is 324 metres (1,063 ft) tall, about the same height as an 81-storey building, and the tallest structure in Paris. Its base is square, measuring 125 metres (410 ft) on each side. During its construction, the Eiffel Tower surpassed the Washington Monument to become the tallest man-made structure in the world, a title it held for 41 years until the Chrysler Building in New York City was finished in 1930. It was the first structure to reach a height of 300 metres. Due to the addition of a broadcasting aerial at the top of the tower in 1957, it is now taller than the Chrysler Building by 5.2 metres (17 ft). Excluding transmitters, the Eiffel Tower is the second tallest free-standing structure in France after the Millau Viaduct.

https://huggingface.co/tasks/summarization
NLP

Classical NLP

Deep Learning-based NLP

Documents

Preprocessing

Language Detection

English

Spanish

Aran

Documents

Preprocessing

Dense Embeddings

obtained via word2vec, doc2vec, GloVe, etc.

Hidden Layers

Output Units

Sentiment

Classification

Entity Extraction

Translation

Topic Modeling

Output

Source: http://blog.aylien.com/leveraging-deep-learning-for-multilingual/
Modern NLP Pipeline

Modern NLP Pipeline

Source: http://mattfortier.me/2017/01/31/nlp-intro-pt-1-overview/
Deep Learning NLP

Documents → Preprocessing → Dense Word Embeddings → Deep Neural Network

Pre-generated Lookup OR Generated in 1st level of NeuralNet

Task / Output
- Classification
- Sentiment Analysis
- Entity Extraction
- Topic Modeling
- Document Similarity

Source: http://mattfortier.me/2017/01/31/nlp-intro-pt-1-overview/
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."

Text Summarization and Information Extraction

• **Key-phrase extraction**
  • extracting key influential phrases from the documents.

• **Topic modeling**
  • Extract various diverse concepts or topics present in the documents, retaining the major themes.

• **Document summarization**
  • Summarize entire text documents to provide a gist that retains the important parts of the whole corpus.

Natural Language Processing (NLP) and Text Mining

- Raw text
- Sentence Segmentation
- Tokenization
- Part-of-Speech (POS)
- Stop word removal
- Stemming / Lemmatization
  - word’s stem: am → am
  - having → hav
- Dependency Parser
- String Metrics & Matching
  - word’s lemma: am → be
  - having → have

Source: Nitin Hardeniya (2015), NLTK Essentials, Packt Publishing; Florian Leitner (2015), Text mining - from Bayes rule to dependency parsing
Text Summarization

Topic Modeling

(a) Single-document or (b) Multi-document, automatic text summarizer
Automatic Text Summarization Approaches

Extractive Text Summarization System

Abstractive Text Summarization System

Hybrid Text Summarization System

Single-sentence and Multi-sentence Text Summarization Operations

Text Summarization Operations

Single-Sentence Operation
- Sentence Compression
- Syntactic Transformation
- Lexical Paraphrasing
- Generalization
- Specification

Multi-Sentence Operation
- Sentence Combination
- Sentence Reordering
- Sentence Selection
- Sentence Clustering

PEGASUS:
Pre-training with Extracted Gap-sentences for Abstractive Summarization

Topic Modeling
Topic Model in Bioinformatics

Topic Modeling

Topic Modeling (Unsupervised Learning) vs. Text Classification (Supervised Learning)

Topic Modeling
Term Document Matrix to
Topic Distribution

Term Document Matrix

<table>
<thead>
<tr>
<th></th>
<th>Doc-1</th>
<th>Doc-2</th>
<th>Doc-3</th>
<th>Doc-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Term-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Term-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Term-3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Term-4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Word Assignment to Topics

<table>
<thead>
<tr>
<th></th>
<th>Topic-1</th>
<th>Topic-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Term-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Term-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Term-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Term-4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Topic Importance

<table>
<thead>
<tr>
<th></th>
<th>Topic-1</th>
<th>Topic-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Topic-2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Topic Distribution Across Documents

<table>
<thead>
<tr>
<th></th>
<th>Doc-1</th>
<th>Doc-2</th>
<th>Doc-3</th>
<th>Doc-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Topic-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

m*m Matrix
m*n Singular Matrix
n*n Diagonal Matrix
n*m Singular Matrix

Topic Modeling
Latent Dirichlet Allocation (LDA)

$D$ documents
$N$ words
$K$ topics

Latent Dirichlet Allocation

David M. Blei
Computer Science Division
University of California
Berkeley, CA 94720, USA

Andrew Y. Ng
Computer Science Department
Stanford University
Stanford, CA 94305, USA

Michael I. Jordan
Computer Science Division and Department of Statistics
University of California
Berkeley, CA 94720, USA

Editor: John Lafferty

Abstract

We describe latent Dirichlet allocation (LDA), a generative probabilistic model for collections of discrete data such as text corpora. LDA is a three-level hierarchical Bayesian model, in which each item of a collection is modeled as a finite mixture over an underlying set of topics. Each topic is, in turn, modeled as an infinite mixture over an underlying set of topic probabilities. In the context of text modeling, the topic probabilities provide an explicit representation of a document. We present efficient approximate inference techniques based on variational methods and an EM algorithm for empirical Bayes parameter estimation. We report results in document modeling, text classification, and collaborative filtering, comparing to a mixture of unigrams model and the probabilistic LSI model.

Topic Modeling Using Latent Dirichlet allocation (LDA)

Topic Modeling Technique

The Generative Process of Latent Dirichlet Allocation (LDA)

Topic Visualization as Word Clouds

LDAvis: Gensim Topic Model Visualization

BERTopic
Neural topic modeling with a class-based TF-IDF procedure


https://github.com/MaartenGr/BERTopic
gensim

Gensim is a FREE Python library

- Scalable statistical semantics
- Analyze plain-text documents for semantic structure
- Retrieve semantically similar documents

from gensim import corpora, models, similarities

# Load corpus iterator from a Matrix Market file on disk.
corpus = corpora.MmCorpus('/path/to/corpus.mm')

# Initialize Latent Semantic Indexing with 200 dimensions.
lsi = models.LsiModel(corpus, num_topics=200)

# Convert another corpus to the latent space and index it.
index = similarities.MatrixSimilarity(lsi[another_corpus])

# Compute similarity of a query vs. indexed documents
sims = index[query]

https://radimrehurek.com/gensim/
spaCy

Industrial-Strength Natural Language Processing in Python

Fastest in the world

spaCy excels at large-scale information extraction tasks. It's written from the ground up in carefully memory-managed Cython. Independent research has confirmed that spaCy is the fastest in the world. If your application needs to process entire web dumps, spaCy is the library you want to be using.

Get things done

spaCy is designed to help you do real work — to build real products, or gather real insights. The library respects your time, and tries to avoid wasting it. It's easy to install, and its API is simple and productive. I like to think of spaCy as the Ruby on Rails of Natural Language Processing.

Deep learning

spaCy is the best way to prepare text for deep learning. It interoperates seamlessly with TensorFlow, Keras, Scikit-Learn, Gensim and the rest of Python's awesome AI ecosystem. spaCy helps you connect the statistical models trained by these libraries to the rest of your application.

https://spacy.io/
Python in Google Colab (Python101)

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

Text Summarization with Gensim Summarization:
https://radimrehurek.com/gensim/auto_examples/tutorials/run_summarization.html

```python
from pprint import pprint as print
from gensim.summarization import summarize

text = {
    "Thomas A. Anderson is a man living two lives. By day he is an 'average computer programmer and by night a hacker known as Neo. Neo has always questioned his reality, but the truth is far beyond his imagination. Neo finds himself targeted by the 'police when he is contacted by Morpheus, a legendary computer 'hacker branded a terrorist by the government. Morpheus awakens Neo to the real world, a ravaged wasteland where most of 'humanity have been captured by a race of machines that live 'off of the humans' body heat and electrochemical energy and 'who imprison their minds within an artificial reality known as 'the Matrix. As a rebel against the machines, Neo must return to "'the Matrix and confront the agents: super-powerful computer "programs devoted to snuffing out Neo and the entire human "rebellion."
}
print(text)
```
Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FE6DnGvWFUbeO4zJ1zTunjMqf2RkCrT

https://tinyurl.com/aintpupython101
# NLP Benchmark Datasets

<table>
<thead>
<tr>
<th>Task</th>
<th>Dataset</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WMT 2014 EN-FR</td>
<td></td>
</tr>
<tr>
<td>Text Summarization</td>
<td>CNN/DM</td>
<td><a href="https://cs.nyu.edu/~kcho/DMQA/">https://cs.nyu.edu/~kcho/DMQA/</a></td>
</tr>
<tr>
<td></td>
<td>Newsroom</td>
<td><a href="https://summarie.es/">https://summarie.es/</a></td>
</tr>
<tr>
<td></td>
<td>Gigaword</td>
<td><a href="https://catalog.ldc.upenn.edu/LDC2012T21">https://catalog.ldc.upenn.edu/LDC2012T21</a></td>
</tr>
<tr>
<td>Reading Comprehension Question Answering</td>
<td>ARC</td>
<td><a href="http://data.allenai.org/arc/">http://data.allenai.org/arc/</a></td>
</tr>
<tr>
<td>Question Generation</td>
<td>CNN/CR</td>
<td><a href="http://aclweb.org/anthology/N18-1140">http://aclweb.org/anthology/N18-1140</a></td>
</tr>
<tr>
<td></td>
<td>CNN/DM</td>
<td><a href="https://cs.nyu.edu/~kcho/DMQA/">https://cs.nyu.edu/~kcho/DMQA/</a></td>
</tr>
<tr>
<td></td>
<td>NewsQA</td>
<td><a href="https://datasets.natuura.com/NewsQA">https://datasets.natuura.com/NewsQA</a></td>
</tr>
<tr>
<td></td>
<td>RACE</td>
<td><a href="http://www.qzhexie.com/data/RACE_leaderboard">http://www.qzhexie.com/data/RACE_leaderboard</a></td>
</tr>
<tr>
<td></td>
<td>SQuAD</td>
<td><a href="https://rajpurkar.github.io/SQuAD-explorer/">https://rajpurkar.github.io/SQuAD-explorer/</a></td>
</tr>
<tr>
<td></td>
<td>NarrativeQA</td>
<td><a href="https://github.com/deepmind/narrativeqa">https://github.com/deepmind/narrativeqa</a></td>
</tr>
<tr>
<td></td>
<td>Quasar</td>
<td><a href="https://github.com/bdhingra/quasar">https://github.com/bdhingra/quasar</a></td>
</tr>
<tr>
<td></td>
<td>SearchQA</td>
<td><a href="https://github.com/nyu-dl/SearchQA">https://github.com/nyu-dl/SearchQA</a></td>
</tr>
<tr>
<td>Semantic Parsing</td>
<td>AMR parsing</td>
<td><a href="https://amr.isi.edu/index.html">https://amr.isi.edu/index.html</a></td>
</tr>
<tr>
<td></td>
<td>ATIS (SQL Parsing)</td>
<td><a href="https://github.com/jkkummerfeld/text2sql-data/tree/master/data">https://github.com/jkkummerfeld/text2sql-data/tree/master/data</a></td>
</tr>
<tr>
<td></td>
<td>WikiSQL (SQL Parsing)</td>
<td><a href="https://github.com/salesforce/WikiSQL">https://github.com/salesforce/WikiSQL</a></td>
</tr>
<tr>
<td>Sentiment Analysis</td>
<td>IMDB Reviews</td>
<td><a href="http://ai.stanford.edu/~amaas/data/sentiment/">http://ai.stanford.edu/~amaas/data/sentiment/</a></td>
</tr>
<tr>
<td></td>
<td>SST</td>
<td><a href="https://nlp.stanford.edu/sentiment/index.html">https://nlp.stanford.edu/sentiment/index.html</a></td>
</tr>
<tr>
<td></td>
<td>Yelp Reviews</td>
<td><a href="https://www.yelp.com/dataset/challenge">https://www.yelp.com/dataset/challenge</a></td>
</tr>
<tr>
<td></td>
<td>Subjectivity Dataset</td>
<td><a href="http://www.cs.cornell.edu/people/pabo/movie-review-data/">http://www.cs.cornell.edu/people/pabo/movie-review-data/</a></td>
</tr>
<tr>
<td></td>
<td>DBpedia</td>
<td><a href="https://wiki.dbpedia.org/Datasets">https://wiki.dbpedia.org/Datasets</a></td>
</tr>
<tr>
<td></td>
<td>TREC</td>
<td><a href="https://nlp.stanford.edu/projects/trec/">https://nlp.stanford.edu/projects/trec/</a></td>
</tr>
<tr>
<td></td>
<td>20 NewsGroup</td>
<td><a href="http://qwone.com/~jason/20Newsgroups/">http://qwone.com/~jason/20Newsgroups/</a></td>
</tr>
<tr>
<td>Natural Language Inference</td>
<td>SNLI Corpus</td>
<td><a href="https://nlp.stanford.edu/projects/snli/">https://nlp.stanford.edu/projects/snli/</a></td>
</tr>
<tr>
<td></td>
<td>MultiNLI</td>
<td><a href="https://www.nyu.edu/projects/bowman/multinli/">https://www.nyu.edu/projects/bowman/multinli/</a></td>
</tr>
<tr>
<td></td>
<td>SciTail</td>
<td><a href="http://data.allenai.org/scitail/">http://data.allenai.org/scitail/</a></td>
</tr>
<tr>
<td>Semantic Role Labeling</td>
<td>Proposition Bank</td>
<td><a href="http://propbank.github.io/">http://propbank.github.io/</a></td>
</tr>
<tr>
<td></td>
<td>OneNotes</td>
<td><a href="https://catalog.ldc.upenn.edu/LDC2013T19">https://catalog.ldc.upenn.edu/LDC2013T19</a></td>
</tr>
</tbody>
</table>

Gensim is a FREE Python library

- Scalable statistical semantics
- Analyze plain-text documents for semantic structure
- Retrieve semantically similar documents

```python
>>> from gensim import corpora, models, similarities
>>> # Load corpus iterator from a Matrix Market file on disk.
>>> corpus = corpora.MmCorpus('/path/to/corpus.mm')

>>> # Initialize Latent Semantic Indexing with 200 dimensions.
>>> lsi = models.LsiModel(corpus, num_topics=200)

>>> # Convert another corpus to the latent space and index it.
>>> index = similarities.MatrixSimilarity(lsi[another_corpus])

>>> # Compute similarity of a query vs. indexed documents
>>> sims = index[query]
```

[https://radimrehurek.com/gensim/](https://radimrehurek.com/gensim/)
spaCy

Industrial-Strength Natural Language Processing in Python

Fastest in the world

spaCy excels at large-scale information extraction tasks. It’s written from the ground up in carefully memory-managed Cython. Independent research has confirmed that spaCy is the fastest in the world. If your application needs to process entire web dumps, spaCy is the library you want to be using.

Get things done

spaCy is designed to help you do real work — to build real products, or gather real insights. The library respects your time, and tries to avoid wasting it. It’s easy to install, and its API is simple and productive. I like to think of spaCy as the Ruby on Rails of Natural Language Processing.

Deep learning

spaCy is the best way to prepare text for deep learning. It interoperates seamlessly with TensorFlow, Keras, Scikit-Learn, Gensim and the rest of Python’s awesome AI ecosystem. spaCy helps you connect the statistical models trained by these libraries to the rest of your application.

https://spacy.io/
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
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!

Python in Google Colab (Python101)

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

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

```python
  2 cd notebooks
  3 from install import *
  4 install_requirements()

[3] 1 from util import *
  2 setup_chapter()

[12] 1 text = """Dear Amazon, last week I ordered an Optimus Prime action figure \n  2 from your online store in Germany. Unfortunately, when I opened the package, \n  3 I discovered to my horror that I had been sent an action figure of Megatron \n  4 instead! As a lifelong enemy of the Decepticons, I hope you can understand my \n  5 dilemma. To resolve the issue, I demand an exchange of Megatron for the \n  6 Optimus Prime figure I ordered. Enclosed are copies of my records concerning \n  7 this purchase. I expect to hear from you soon. Sincerely, Rumblebee."
```
Python in Google Colab (Python101)

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

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

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

https://tinyurl.com/aintpupython101
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

```python
# NER: https://huggingface.co/tasks/token-classification
import transformers
pipeline = transformers.pipeline('ner')

print(pipeline("Hello I'm Omar and I live in Zürich."))
```

```
import transformers
pipeline = transformers.pipeline('ner')

print(pipeline("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
`[{"end": 14,  
"entity": "I-PER",  
"index": 5,  
"score": 0.99770516,  
"start": 10,  
"word": "Omar"},  
{"end": 35,  
"entity": "I-LOC",  
"index": 10,  
"score": 0.99868976,  
"start": 29,  
"word": "Zürich"}]`)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4z1zTunjMqf2RkCrT
https://tinyurl.com/aintpupython101
Python in Google Colab (Python101)

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

1. pip install transformers
2. from transformers import pipeline
3. classifier = pipeline("summarization")
4. 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 100,000 people."
5. classifier(text, max_length=30)

No model was supplied, default to shibboleth/distilbert-cnn-12-6 (https://huggingface.co/shibboleth/distilbert-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']

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

https://tinyurl.com/aintpupython101
Bumblebee 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.

https://tinyurl.com/aintpuppython101
Summary

• Text Summarization
  • Extractive Text Summarization
  • Abstractive Text Summarization
    • PEGASUS: Abstractive Summarization

• Topic Models
  • Topic Modeling
  • Latent Dirichlet Allocation (LDA)
  • BERTopic
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.


• Selva Prabhakaran (2020), Topic modeling visualization – How to present the results of LDA models?, https://www.machinelearningplus.com/nlp/topic-modeling-visualization-how-to-present-results-lda-models/

• 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.


• The Super Duper NLP Repo, https://notebooks.quantumstat.com/


• NLP with Transformer, https://github.com/nlp-with-transformers/notebooks

• Min-Yuh Day (2022), Python 101, https://tinyurl.com/aintpupython101