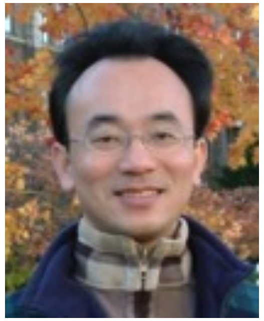
Artificial Intelligence for Text Analytics

Text Summarization and Topic Models

1102AITA07
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

2022-04-26

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

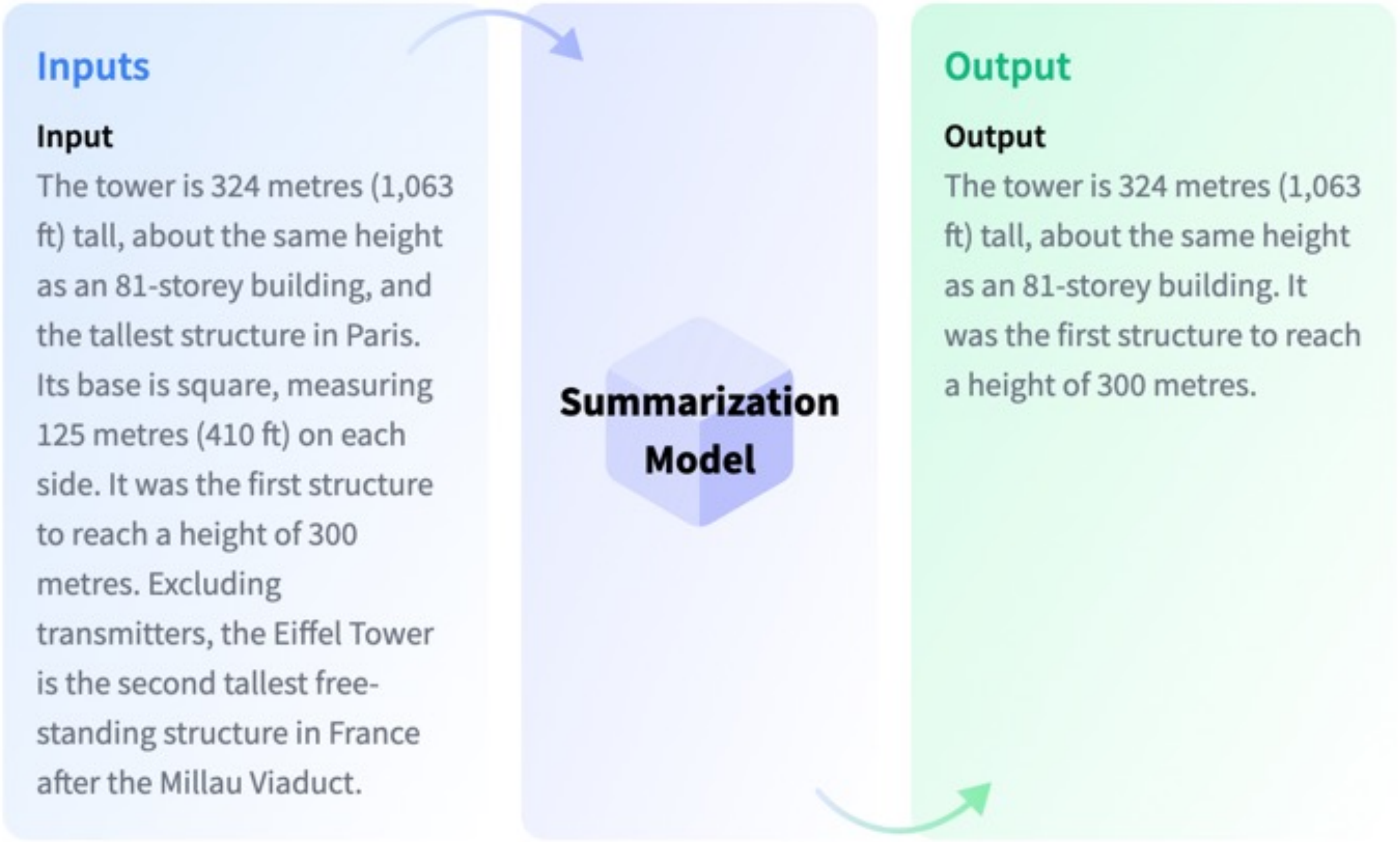
Text Summarization
and
Topic Models

Outline

- Text Summarization
  - Extractive Text Summarization
  - Abstractive Text Summarization
    - PEGASUS: Abstractive Summarization
- Topic Models
  - Topic Modeling
  - Latent Dirichlet Allocation (LDA)
  - BERTopic

Text Summarization

Text Summarization



Inputs

Input

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.

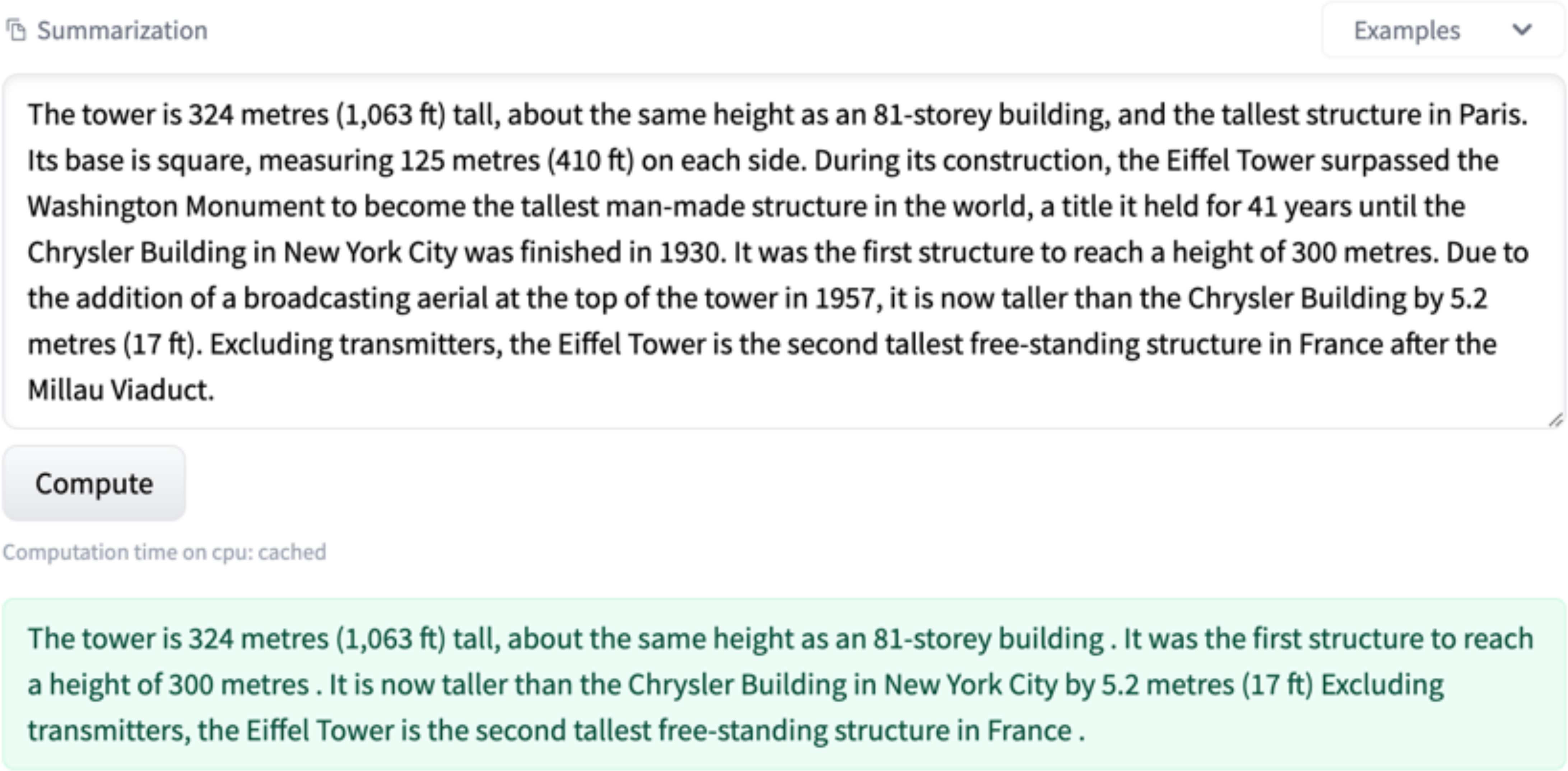
Output

Output

The tower is 324 metres (1,063 ft) tall, about the same height as an 81-storey building. It was the first structure to reach a height of 300 metres.

https://huggingface.co/tasks/summarization

Text Summarization



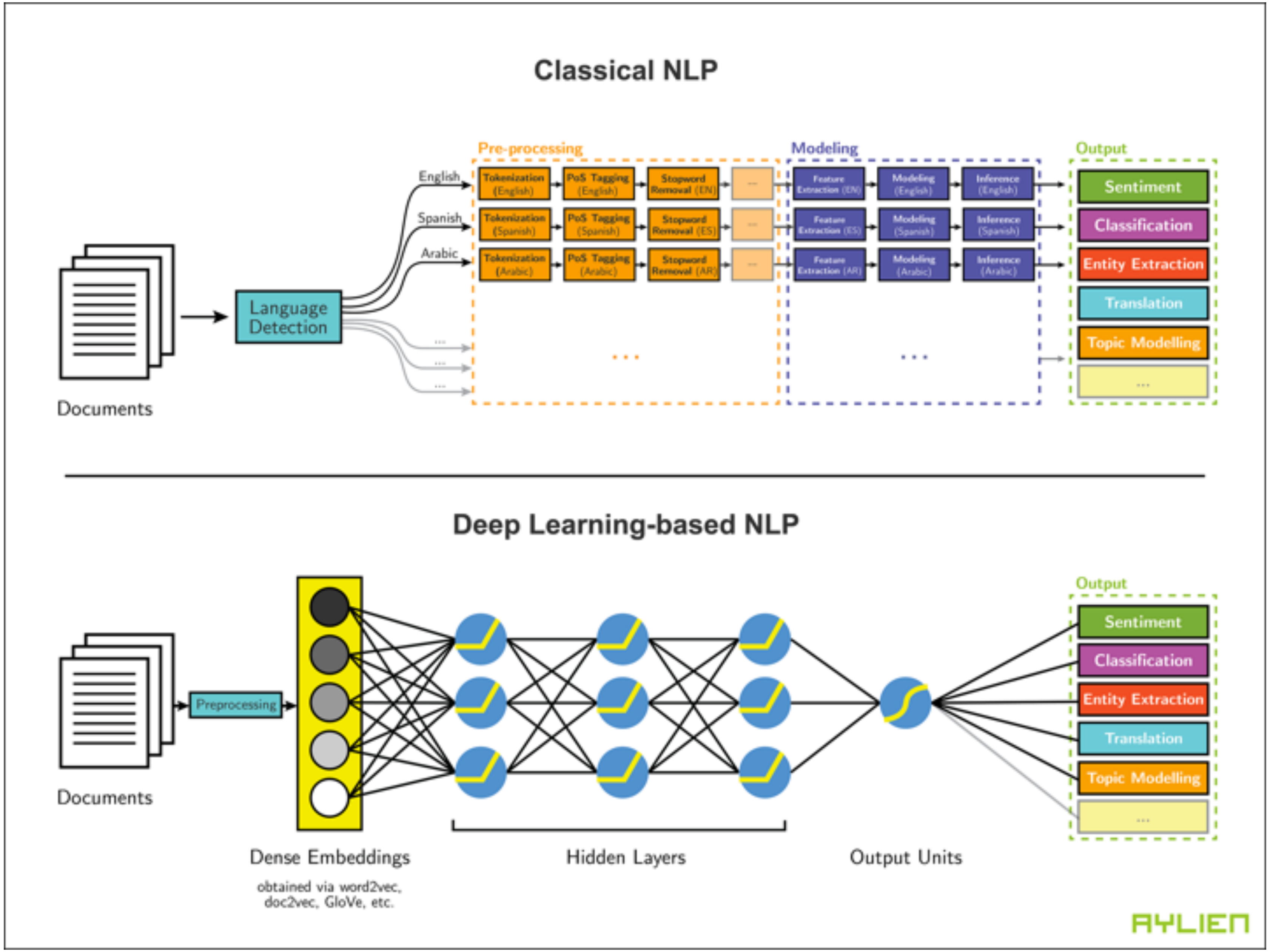
The tower is 324 metres (1,063 ft) tall, about the same height as an 81-storey building, and the tallest structure in Paris. Its base is square, measuring 125 metres (410 ft) on each side. During its construction, the Eiffel Tower surpassed the Washington Monument to become the tallest man-made structure 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.

Computation time on cpu: cached

The tower is 324 metres (1,063 ft) tall, about the same height as an 81-storey building . It was the first structure to reach a height of 300 metres . It is now taller than the Chrysler Building in New York City by 5.2 metres (17 ft) Excluding transmitters, the Eiffel Tower is the second tallest free-standing structure in France .

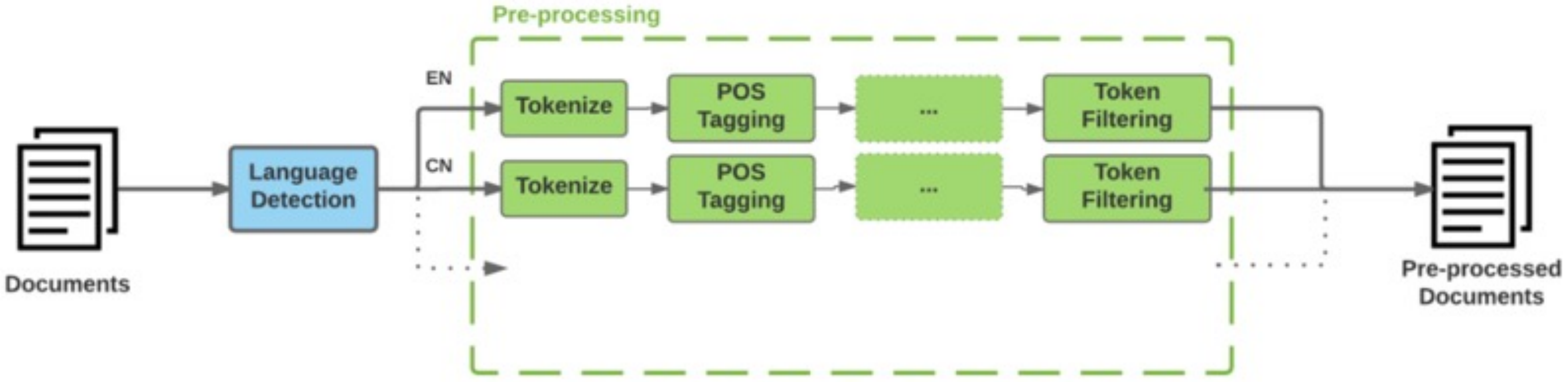
https://huggingface.co/tasks/summarization

NLP



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

Modern NLP Pipeline



Pre-processing

Documents

Language Detection

EN

Tokenize

POS Tagging

...

Token Filtering

CN

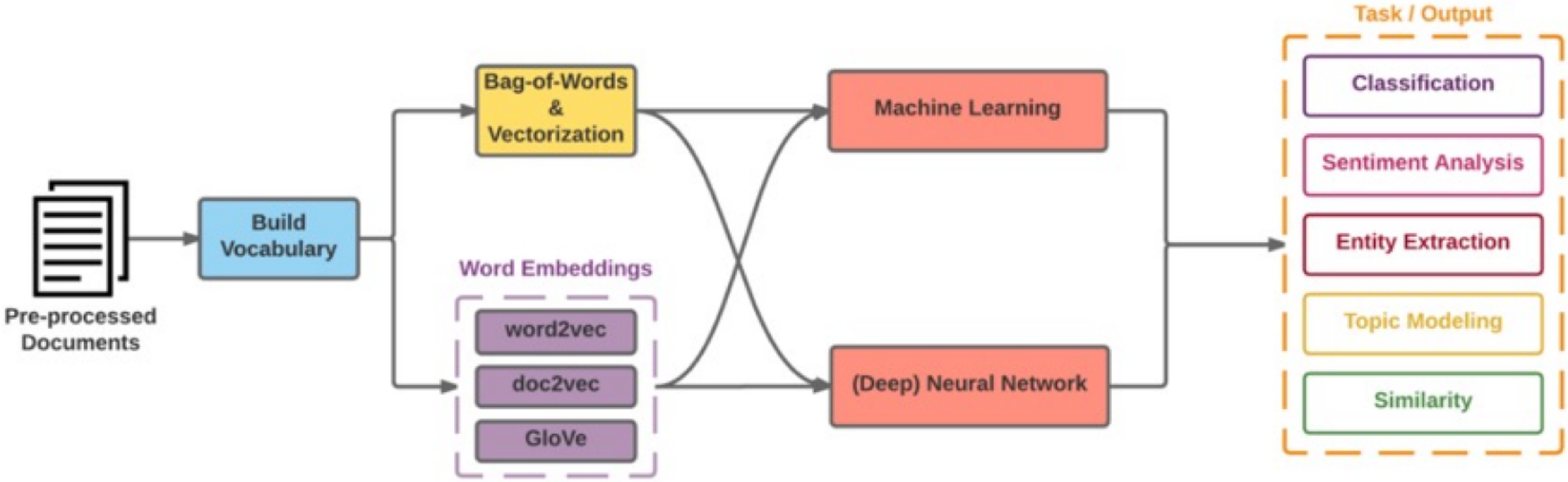
Tokenize

POS Tagging

...

Token Filtering

Pre-processed Documents



Pre-processed Documents

Build Vocabulary

Bag-of-Words & Vectorization

Word Embeddings

word2vec

doc2vec

GloVe

Machine Learning

(Deep) Neural Network

Task / Output

Classification

Sentiment Analysis

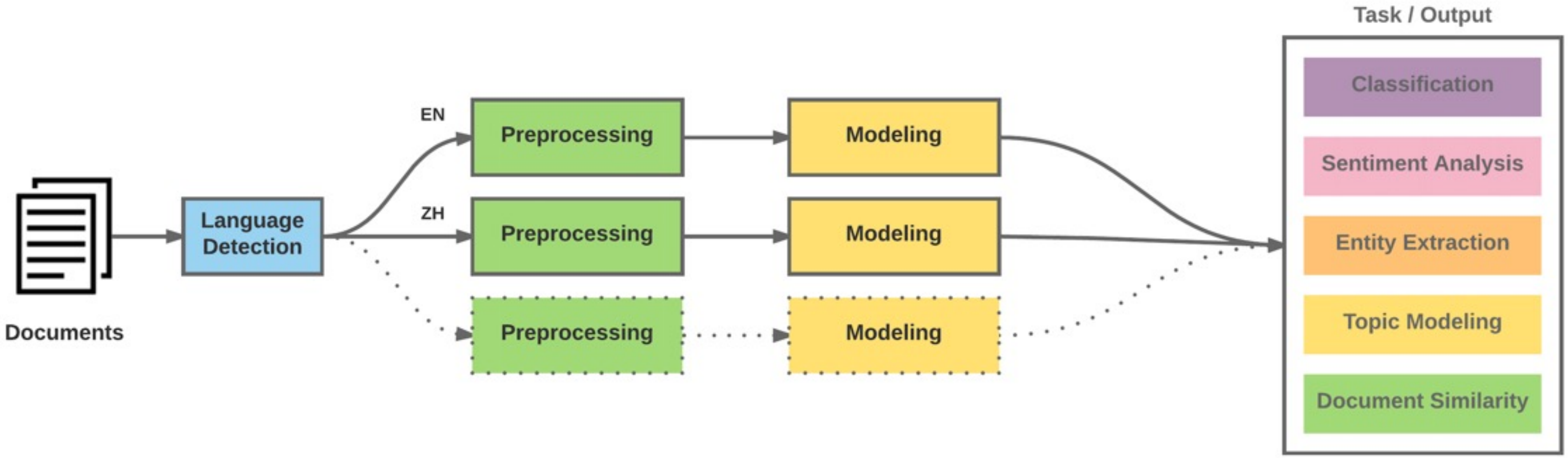
Entity Extraction

Topic Modeling

Similarity

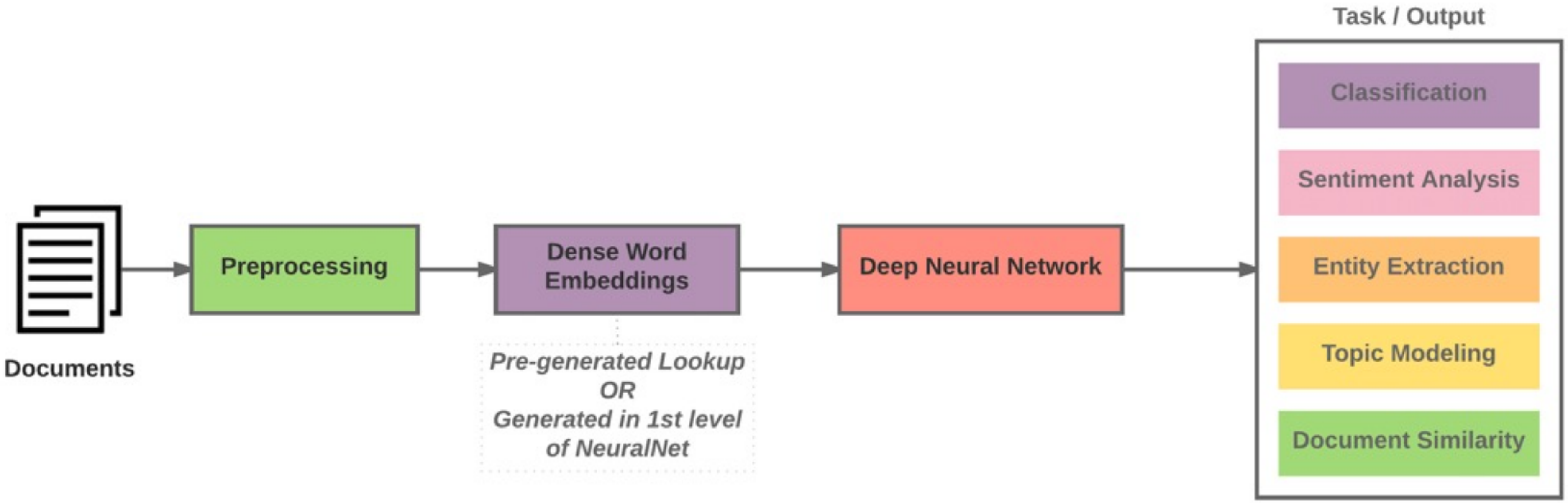
Source: https://github.com/fortiema/talks/blob/master/opendata2016sh/pragmatic-nlp-opendata2016sh.pdf

Modern NLP Pipeline



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

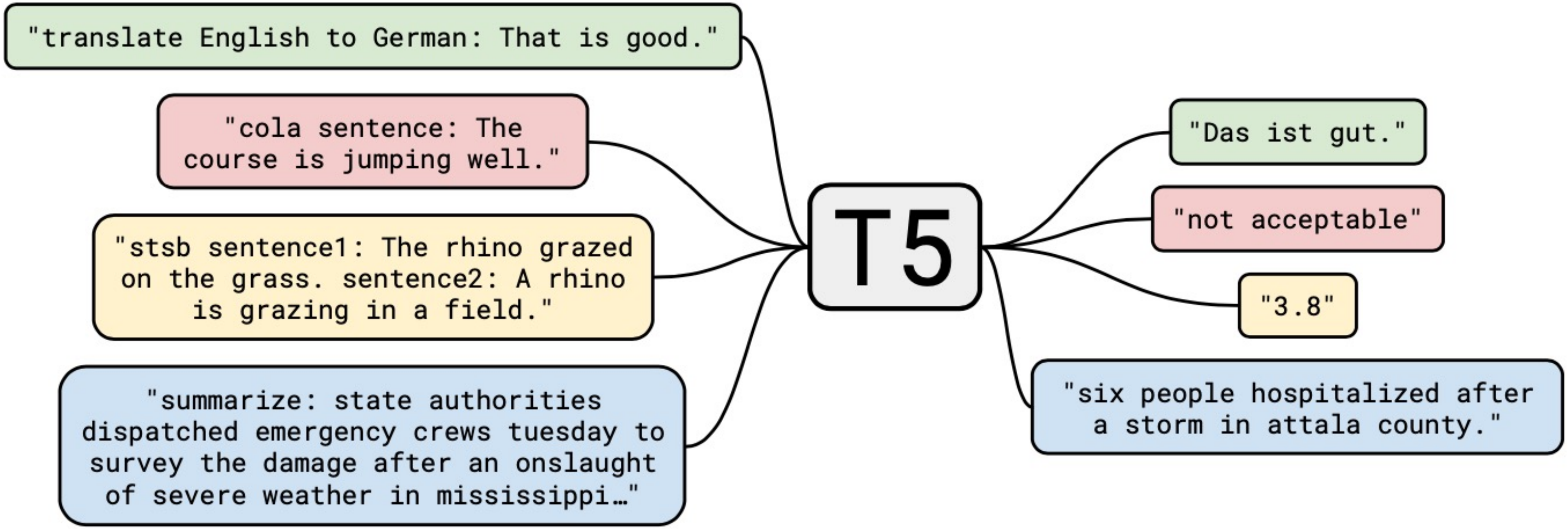
Deep Learning NLP



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."

"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."

Source: JColin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu (2019). "Exploring the limits of transfer learning with a unified text-to-text transformer." arXiv preprint arXiv:1910.10683 (2019).

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.

Source: Dipanjan Sarkar (2019), Text Analytics with Python: A Practitioner's Guide to Natural Language Processing, Second Edition. APress.

Natural Language Processing (NLP) and Text Mining

Raw text

Sentence Segmentation

Tokenization

Part-of-Speech (POS)

Stop word removal

Stemming / Lemmatization

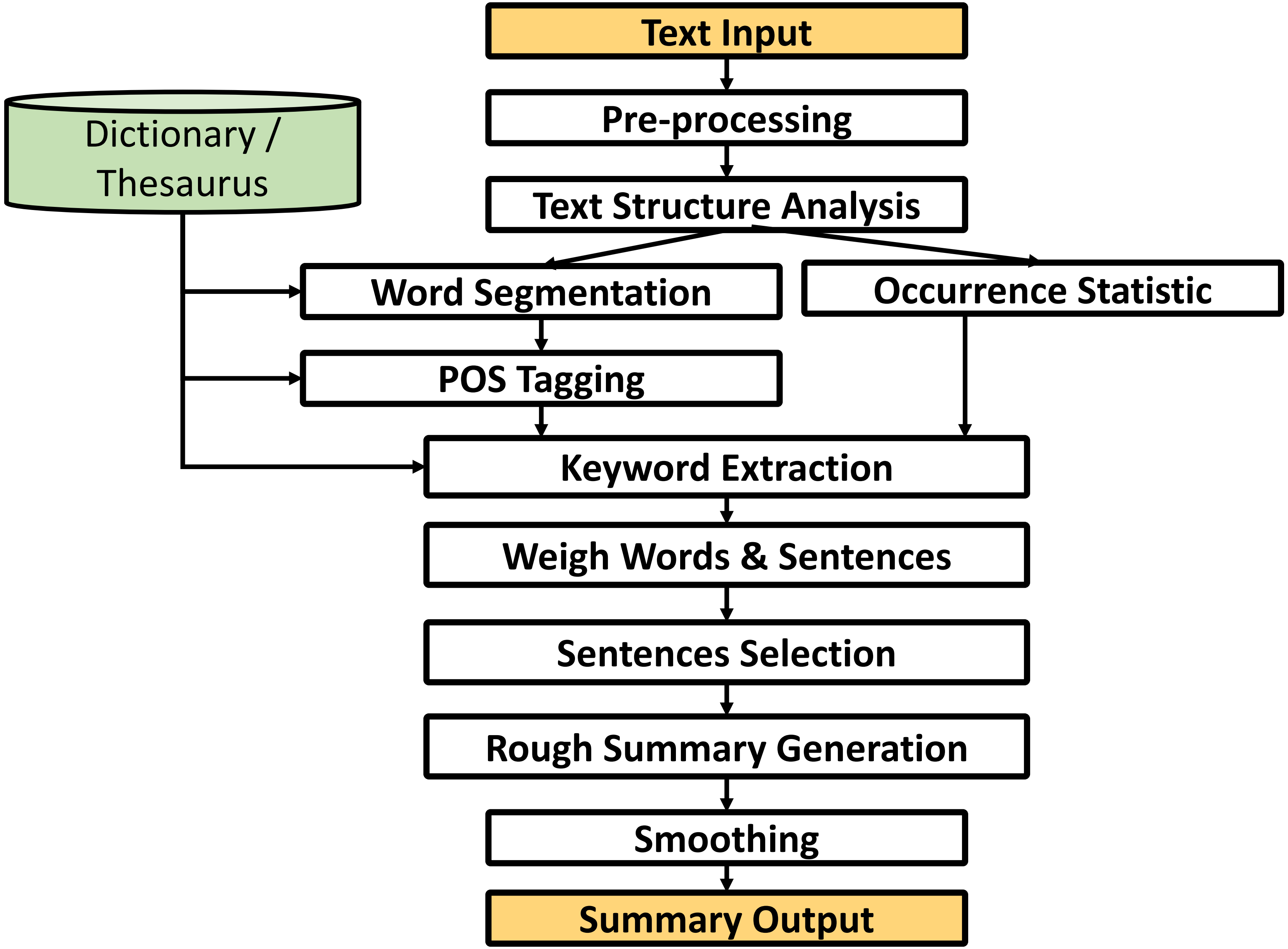
Dependency Parser

String Metrics & Matching

word’s stem	word’s lemma
am → am	am → be
having → hav	having → have

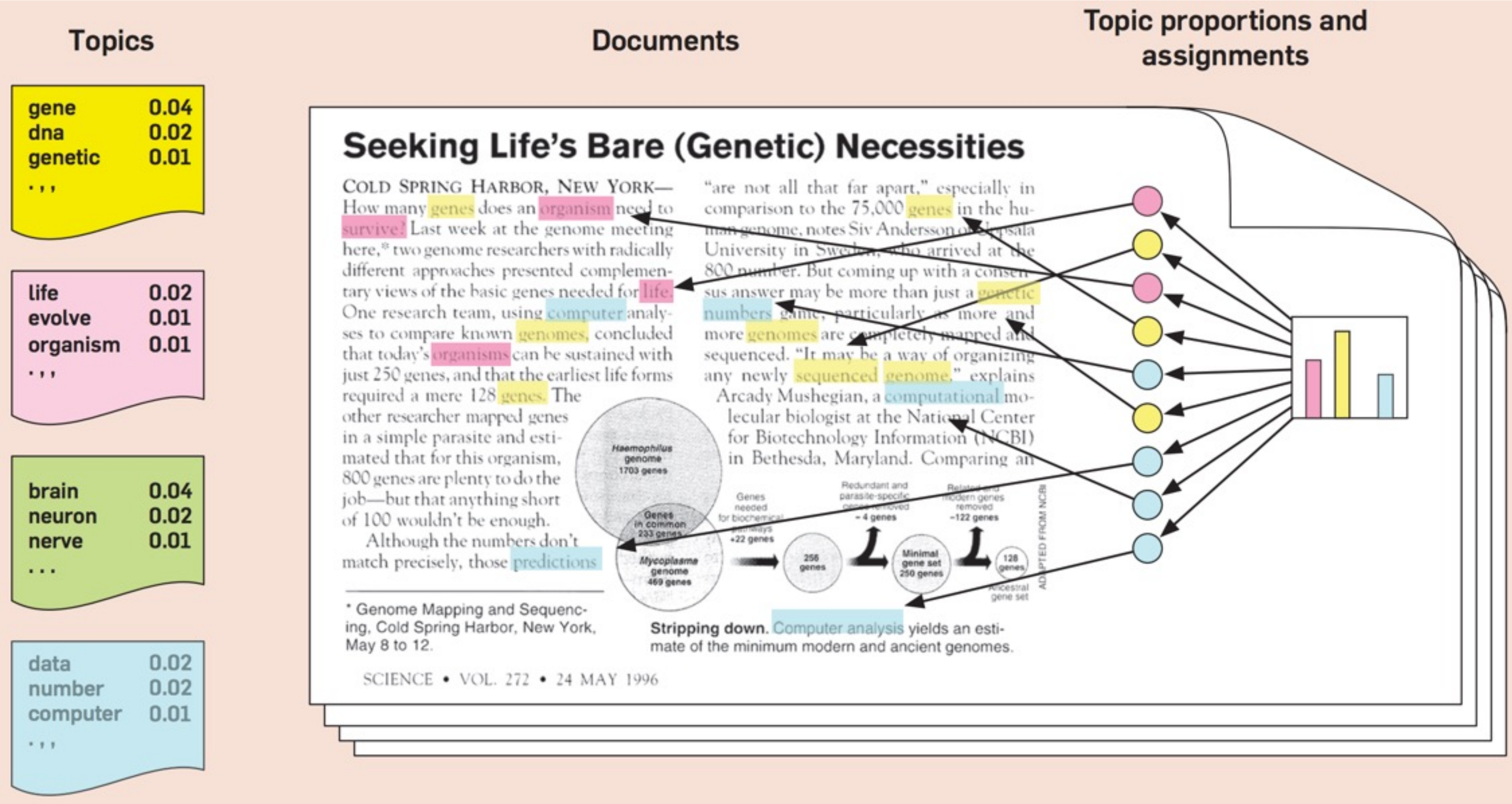
Source: Nitin Hardeniya (2015), NLTK Essentials, Packt Publishing; Florian Leitner (2015), Text mining - from Bayes rule to dependency parsing

Text Summarization



Source: Vishal Gupta and Gurpreet S. Lehal (2009), "A survey of text mining techniques and applications," Journal of emerging technologies in web intelligence, vol. 1, no. 1, pp. 60-76.

Topic Modeling



Topics

gene	0.04
dna	0.02
genetic	0.01
...	

life	0.02
evolve	0.01
organism	0.01
...	

brain	0.04
neuron	0.02
nerve	0.01
...	

data	0.02
number	0.02
computer	0.01
...	

Documents

Topic proportions and assignments

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

Haemophilus genome 1703 genes
Genes in common 233 genes
Mycoplasma genome 469 genes
Genes needed for biochemical pathways +22 genes
256 genes
Redundant and parasite-specific genes removed −4 genes
Minimal gene set 250 genes
Related and modern genes removed −122 genes
128 genes
Ancestral gene set
ADAPTED FROM NCBI

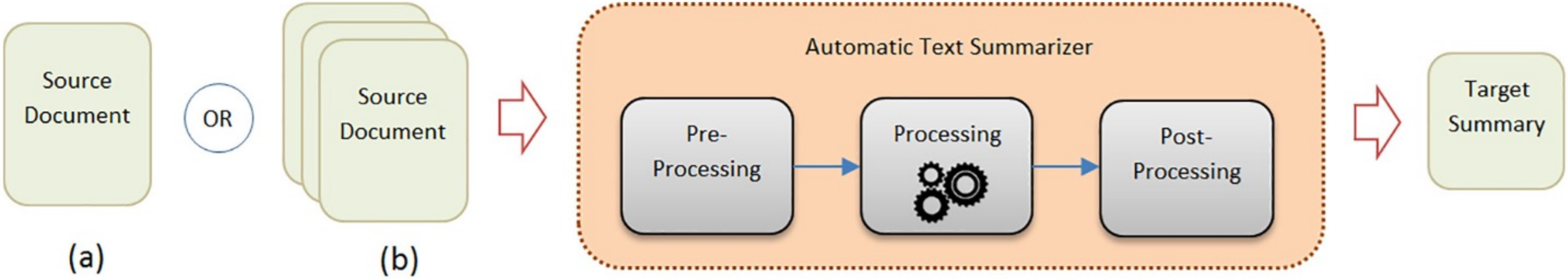
Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

Source: Blei, David M. "Probabilistic topic models." *Communications of the ACM* 55, no. 4 (2012): 77-84.

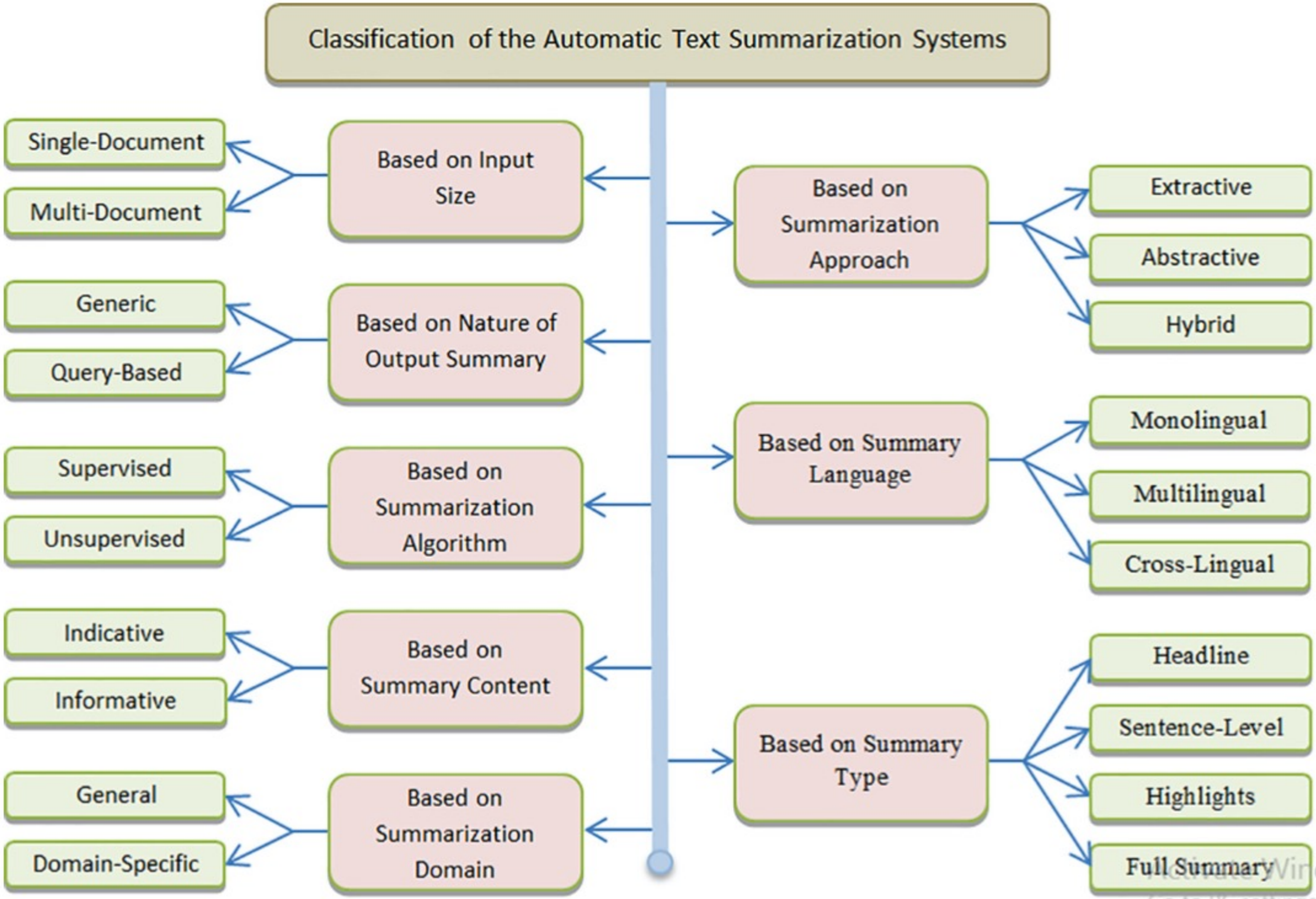
Automatic Text Summarization



(a) Single-document or (b) Multi-document, automatic text summarizer

Source: Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed (2021). "Automatic text summarization: A comprehensive survey." Expert Systems with Applications 165 (2021): 113679.

Classification of Automatic Text Summarization Systems

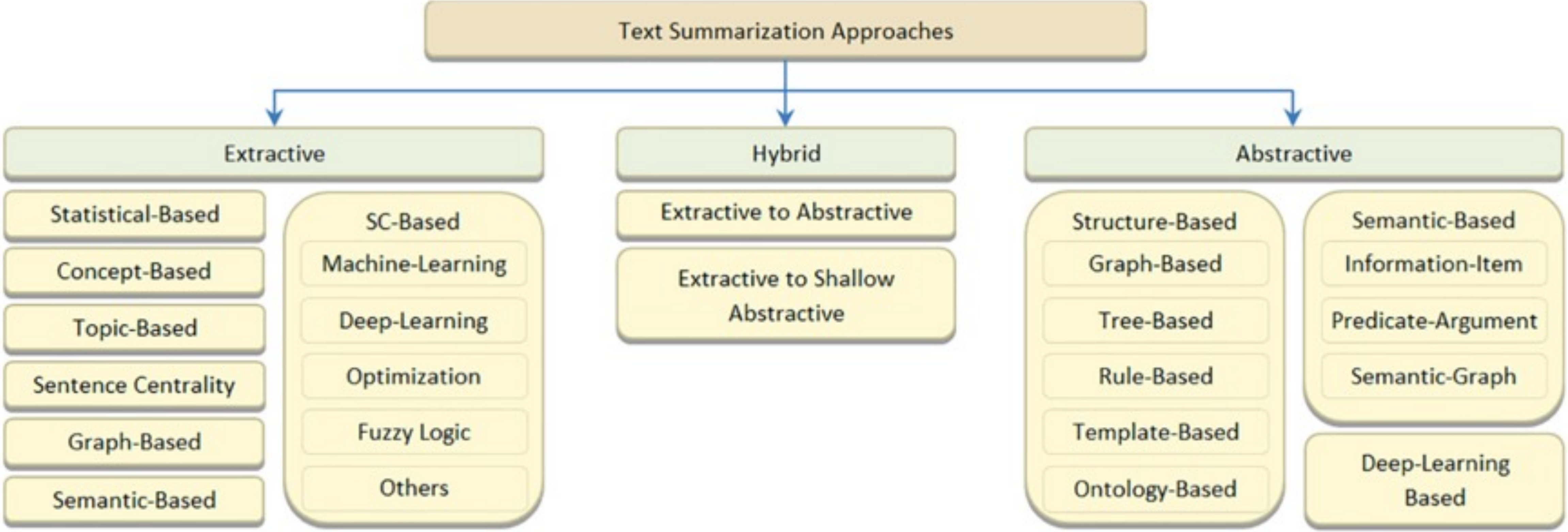


Classification of the Automatic Text Summarization Systems

- Based on Input Size
  - Single-Document
  - Multi-Document
- Based on Nature of Output Summary
  - Generic
  - Query-Based
- Based on Summarization Algorithm
  - Supervised
  - Unsupervised
- Based on Summary Content
  - Indicative
  - Informative
- Based on Summarization Domain
  - General
  - Domain-Specific
- Based on Summarization Approach
  - Extractive
  - Abstractive
  - Hybrid
- Based on Summary Language
  - Monolingual
  - Multilingual
  - Cross-Lingual
- Based on Summary Type
  - Headline
  - Sentence-Level
  - Highlights
  - Full Summary

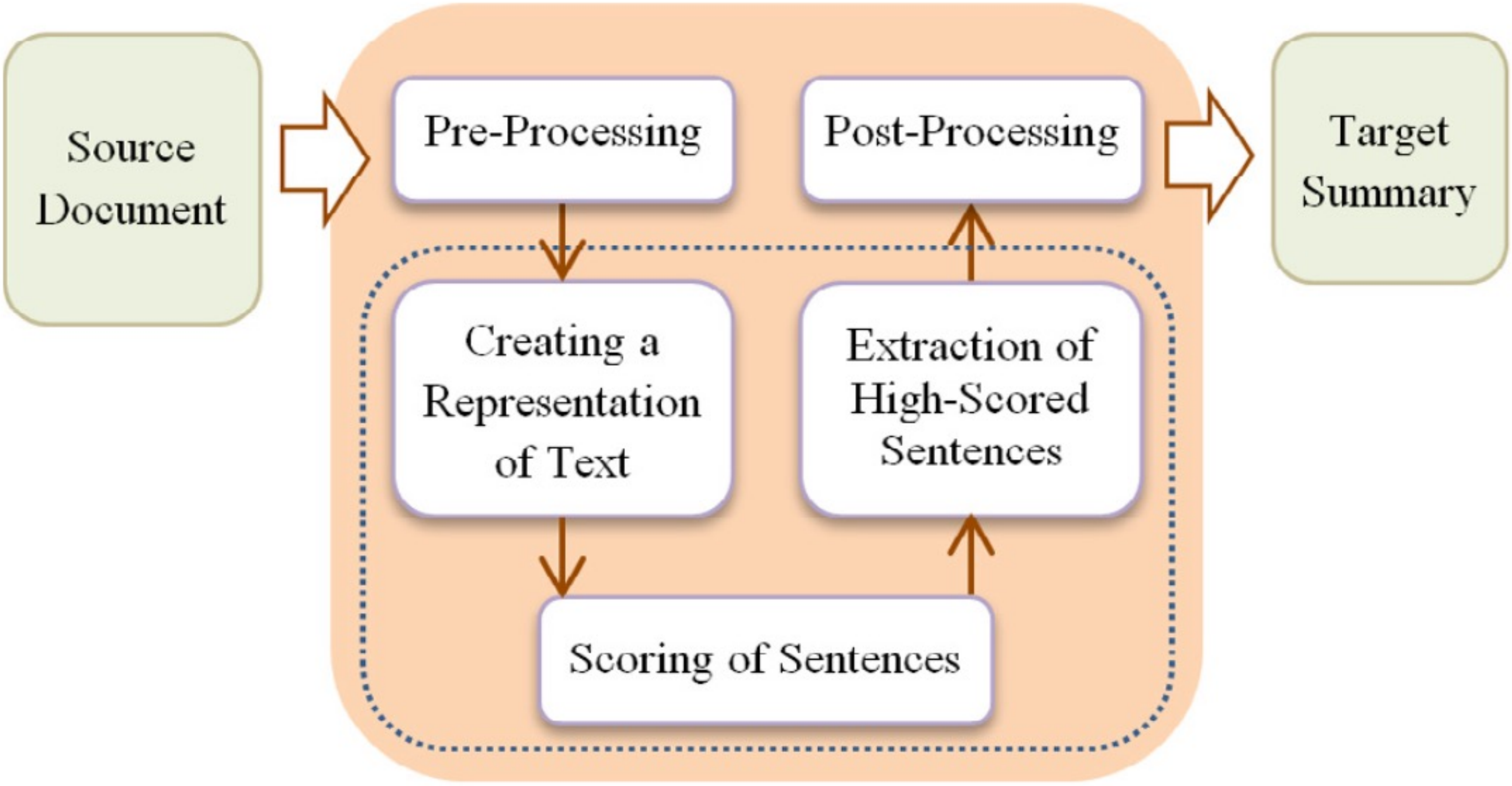
Source: Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed (2021). "Automatic text summarization: A comprehensive survey." Expert Systems with Applications 165 (2021): 113679.

Automatic Text Summarization Approaches



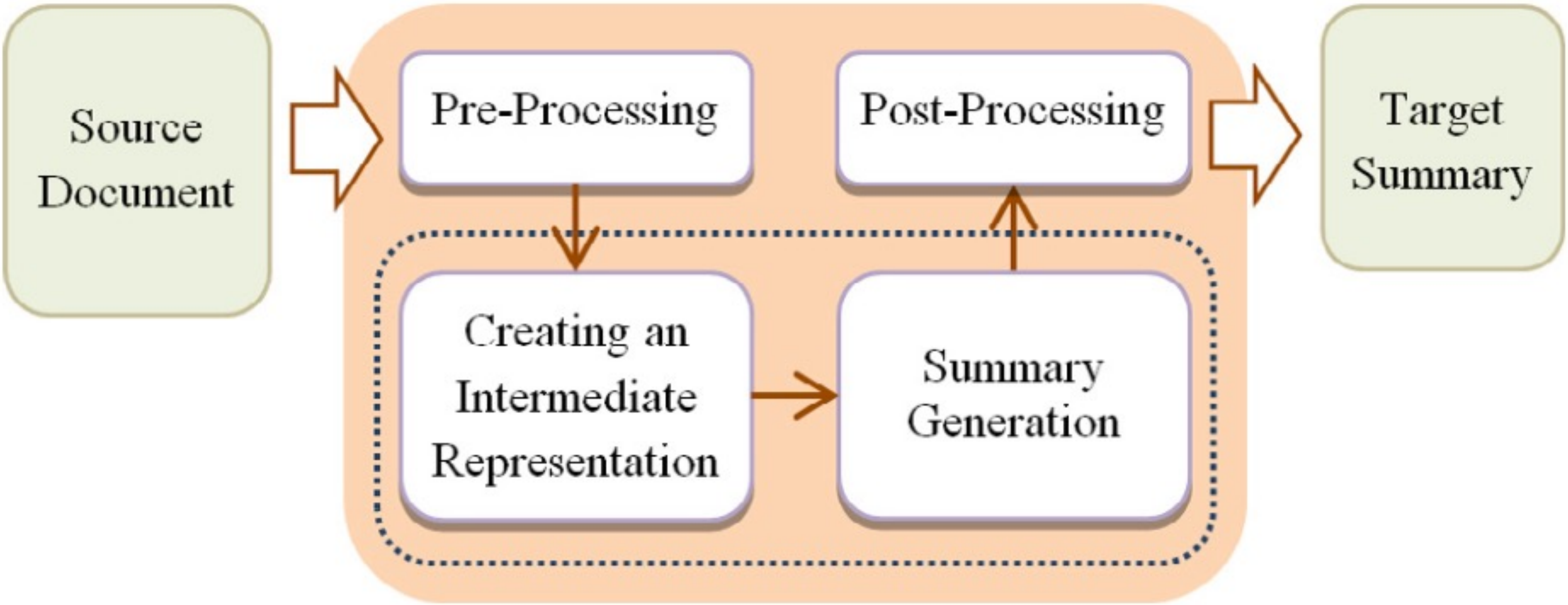
Source: Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed (2021). "Automatic text summarization: A comprehensive survey." Expert Systems with Applications 165 (2021): 113679.

Extractive Text Summarization System



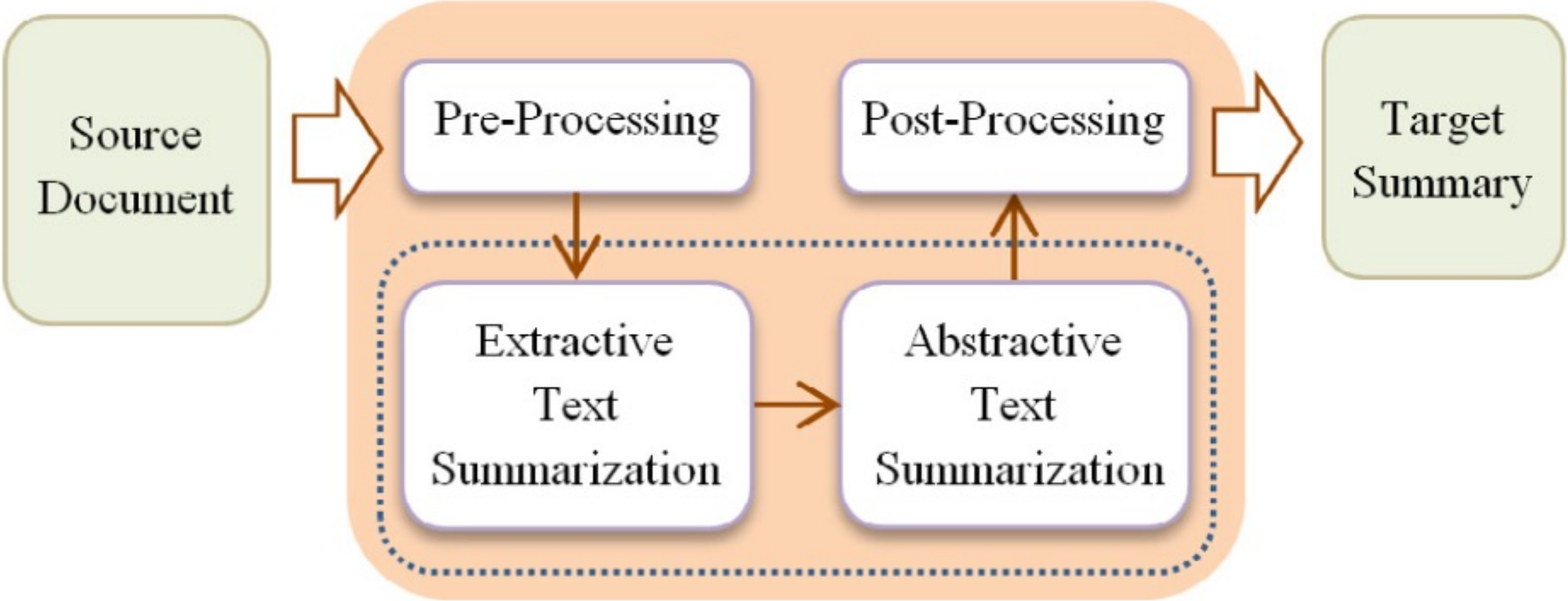
Source: Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed (2021). "Automatic text summarization: A comprehensive survey." Expert Systems with Applications 165 (2021): 113679.

Abstractive Text Summarization System



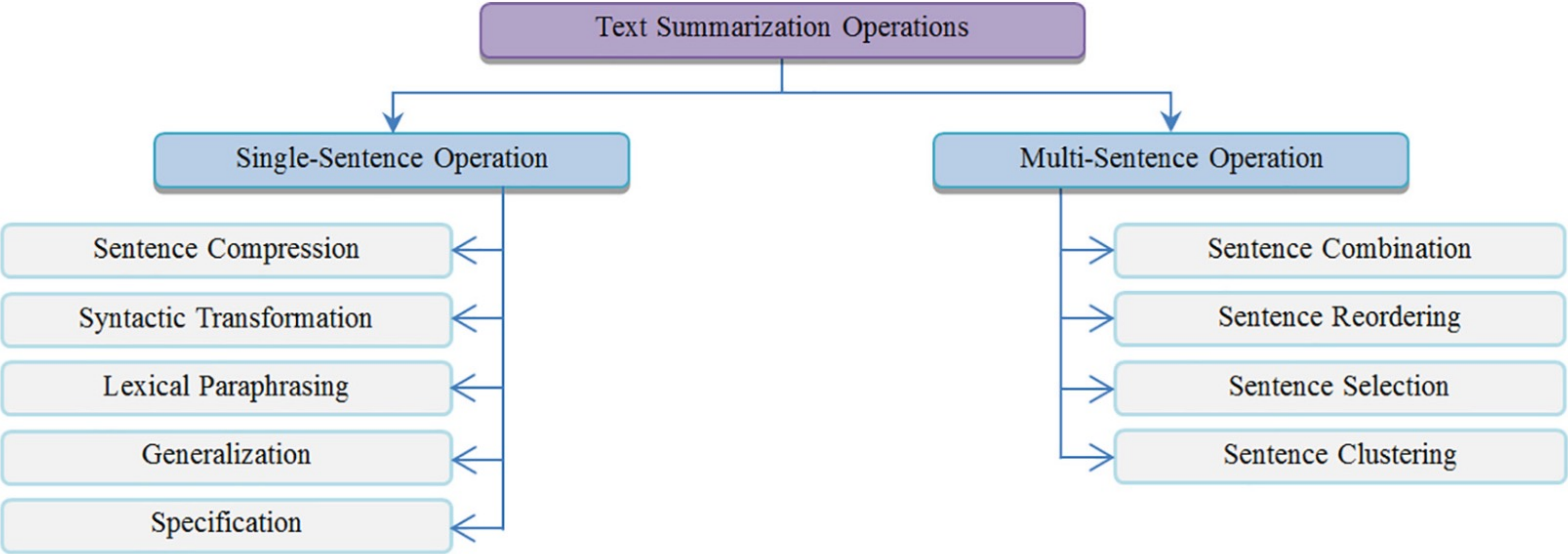
Source: Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed (2021). "Automatic text summarization: A comprehensive survey." Expert Systems with Applications 165 (2021): 113679.

Hybrid Text Summarization System



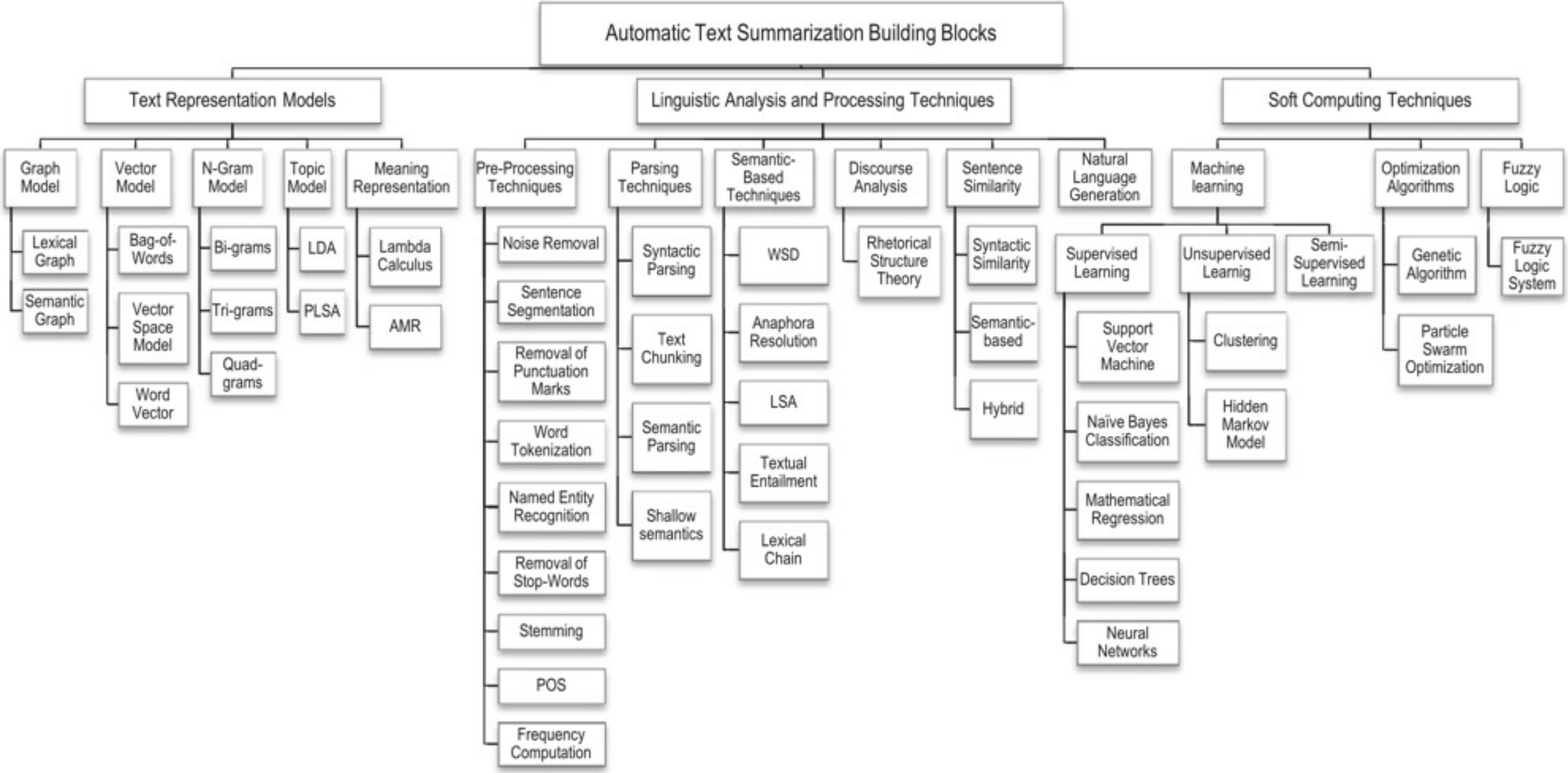
Source: Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed (2021). "Automatic text summarization: A comprehensive survey." Expert Systems with Applications 165 (2021): 113679.

Single-sentence and Multi-sentence Text Summarization Operations



Source: Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed (2021). "Automatic text summarization: A comprehensive survey." Expert Systems with Applications 165 (2021): 113679.

Automatic Text Summarization Building Blocks



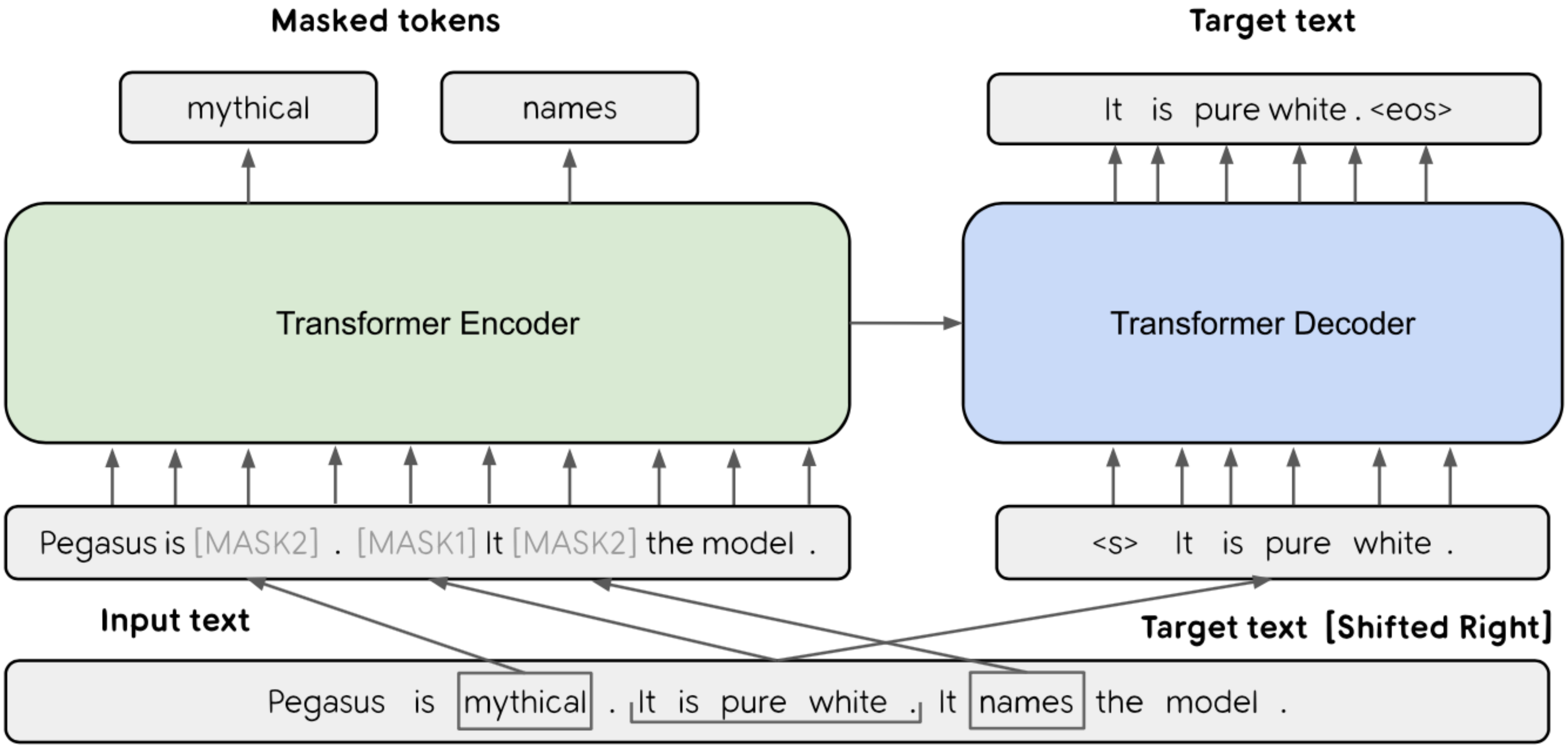
Automatic Text Summarization Building Blocks

- Text Representation Models
  - Graph Model
    - Lexical Graph
    - Semantic Graph
  - Vector Model
    - Bag-of-Words
    - Vector Space Model
    - Word Vector
  - N-Gram Model
    - Bi-grams
    - Tri-grams
    - Quad-grams
  - Topic Model
    - LDA
    - PLSA
  - Meaning Representation
    - Lambda Calculus
    - AMR
- Linguistic Analysis and Processing Techniques
  - Pre-Processing Techniques
    - Noise Removal
    - Sentence Segmentation
    - Removal of Punctuation Marks
    - Word Tokenization
    - Named Entity Recognition
    - Removal of Stop-Words
    - Stemming
    - POS
    - Frequency Computation
  - Parsing Techniques
    - Syntactic Parsing
    - Text Chunking
    - Semantic Parsing
    - Shallow semantics
  - Semantic-Based Techniques
    - WSD
    - Anaphora Resolution
    - LSA
    - Textual Entailment
    - Lexical Chain
  - Discourse Analysis
    - Rhetorical Structure Theory
  - Sentence Similarity
    - Syntactic Similarity
    - Semantic-based
    - Hybrid
  - Natural Language Generation
- Soft Computing Techniques
  - Machine learning
    - Supervised Learning
      - Support Vector Machine
      - Naïve Bayes Classification
      - Mathematical Regression
      - Decision Trees
      - Neural Networks
    - Unsupervised Learnig
      - Clustering
      - Hidden Markov Model
    - Semi-Supervised Learning
  - Optimization Algorithms
    - Genetic Algorithm
    - Particle Swarm Optimization
  - Fuzzy Logic
    - Fuzzy Logic System

Source: Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed (2021). "Automatic text summarization: A comprehensive survey." Expert Systems with Applications 165 (2021): 113679.

PEGASUS:

Pre-training with Extracted Gap-sentences for Abstractive Summarization

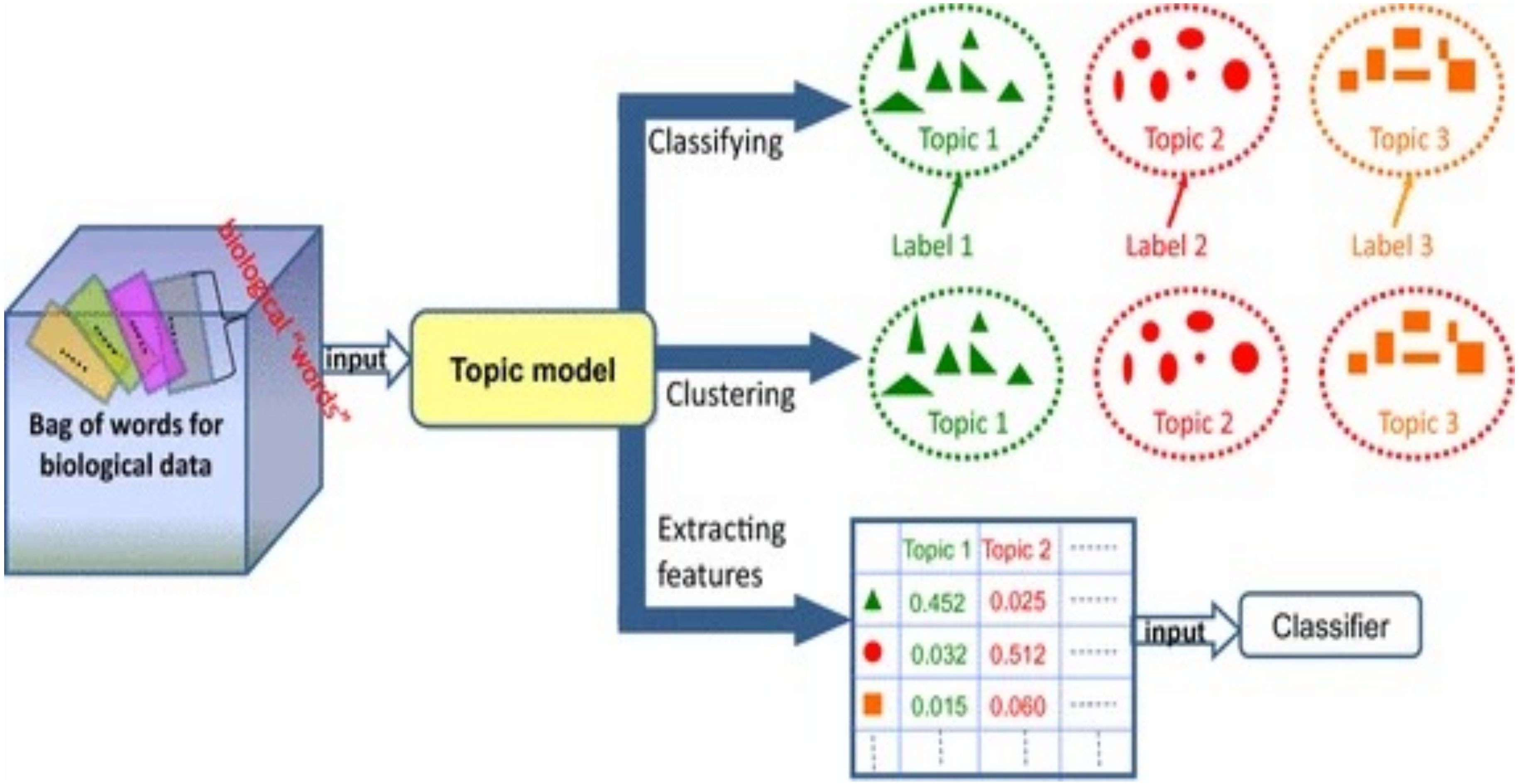


Masked tokens
Target text
mythical
names
It is pure white . <eos>
Transformer Encoder
Transformer Decoder
Pegasus is [MASK2] . [MASK1] It [MASK2] the model .
<s> It is pure white .
Input text
Target text [Shifted Right]
Pegasus is mythical . It is pure white . It names the model .

Source: Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu (2020). "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization." In International Conference on Machine Learning, pp. 11328-11339. PMLR, 2020.

Topic Modeling

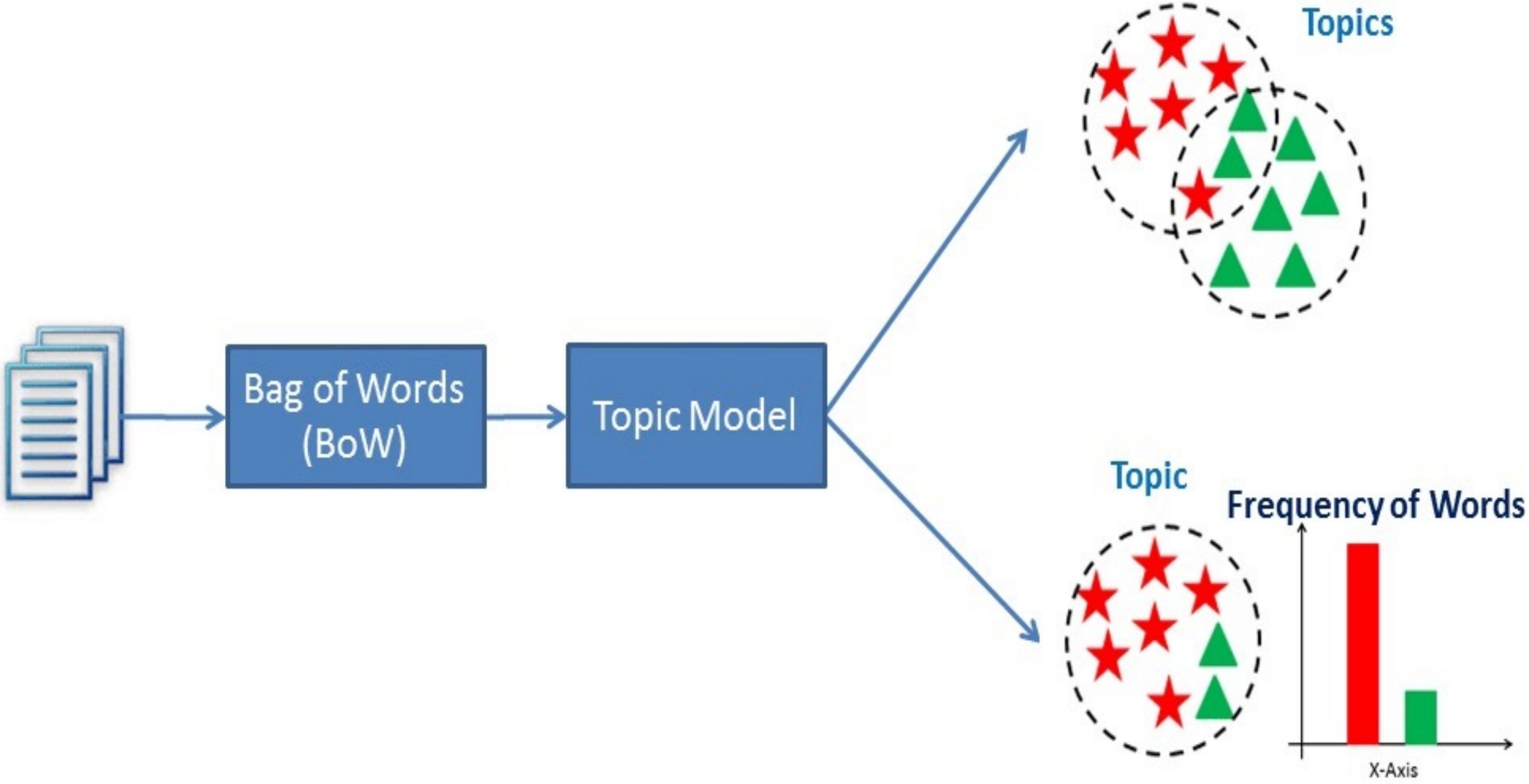
Topic Model in Bioinformatics



	Topic 1	Topic 2	······
▲	0.452	0.025	······
●	0.032	0.512	······
■	0.015	0.060	······
⋮	⋮	⋮	⋮

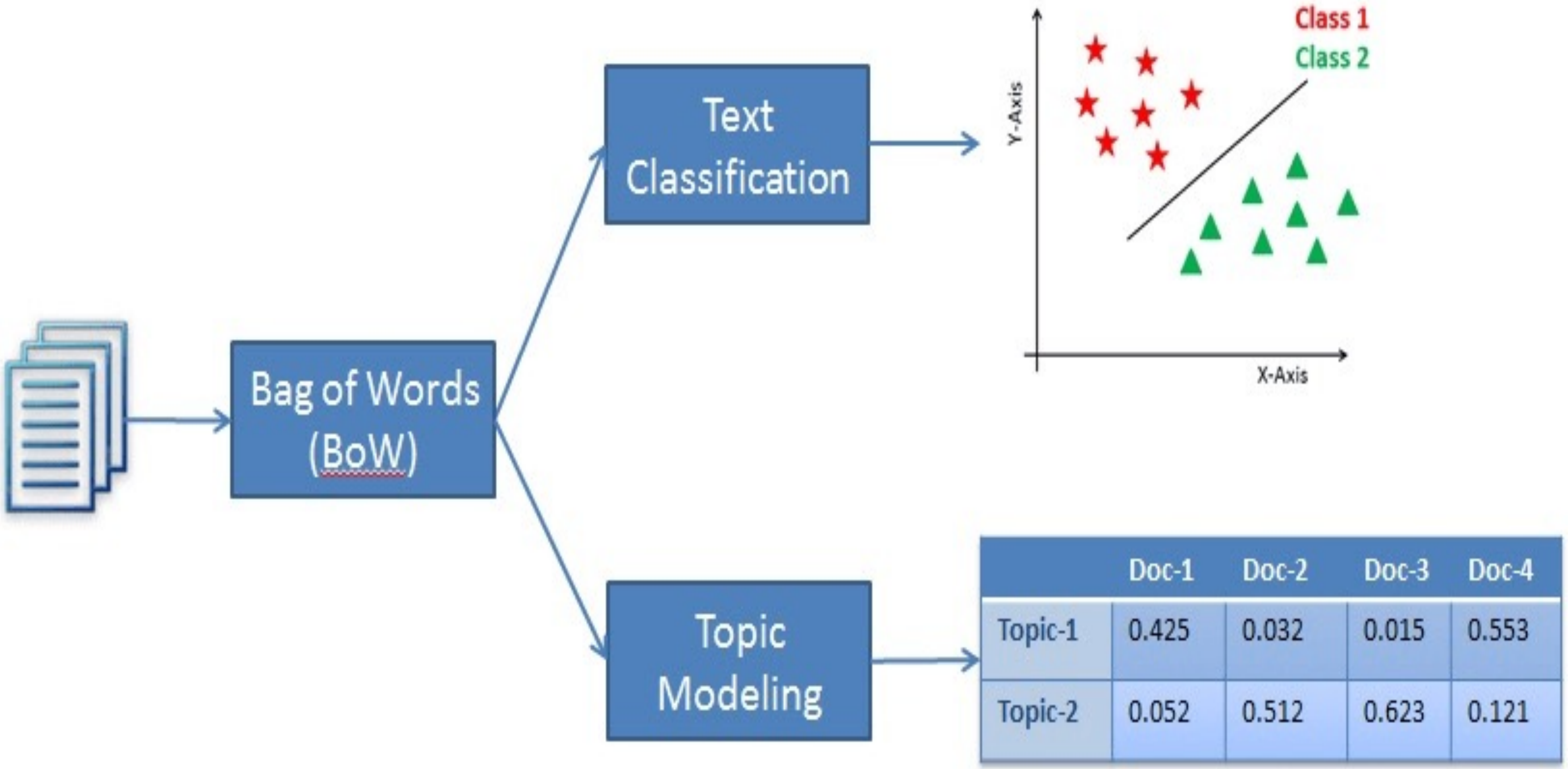
Source: Liu, Lin, Lin Tang, Wen Dong, Shaowen Yao, and Wei Zhou. "An overview of topic modeling and its current applications in bioinformatics." SpringerPlus 5, no. 1 (2016): 1608.

Topic Modeling



Source: Avinash Navlani (2018), Latent Semantic Analysis using Python, https://www.datacamp.com/community/tutorials/discovering-hidden-topics-python

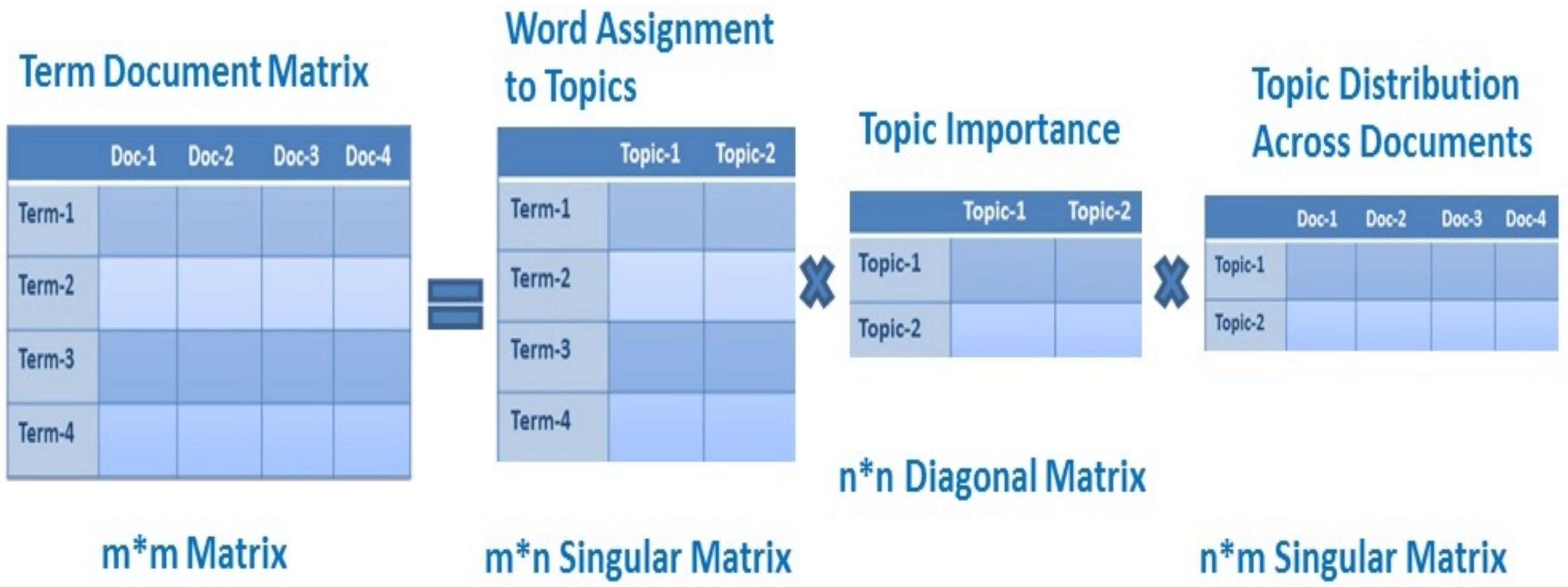
Topic Modeling (Unsupervised Learning) vs. Text Classification (Supervised Learning)



	Doc-1	Doc-2	Doc-3	Doc-4
Topic-1	0.425	0.032	0.015	0.553
Topic-2	0.052	0.512	0.623	0.121

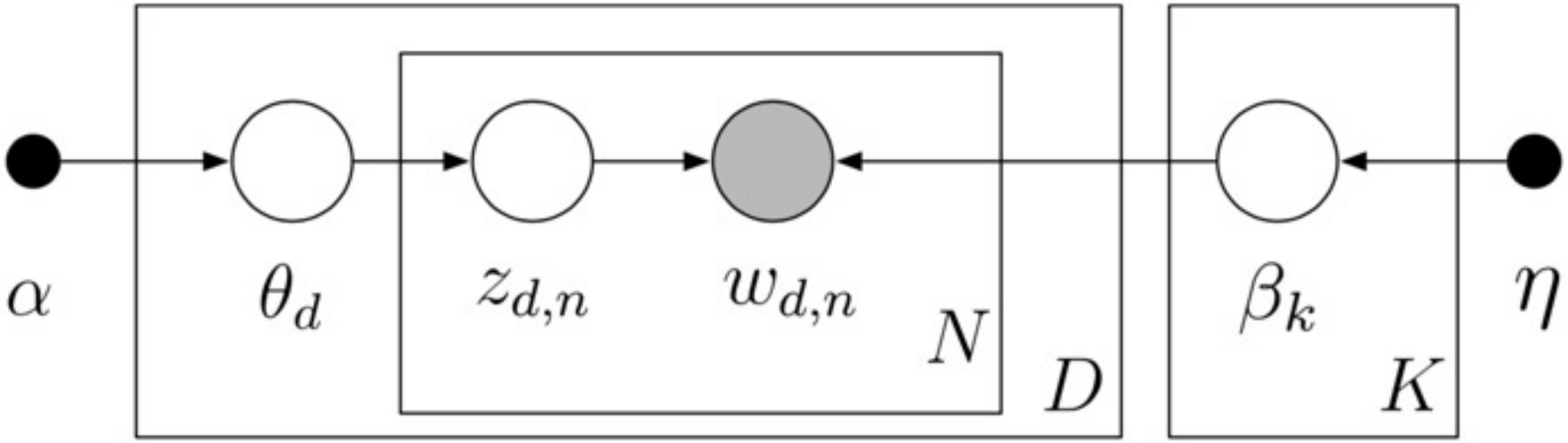
Source: Avinash Navlani (2018), Latent Semantic Analysis using Python, https://www.datacamp.com/community/tutorials/discovering-hidden-topics-python

Topic Modeling
Term Document Matrix to Topic Distribution



Source: Avinash Navlani (2018), Latent Semantic Analysis using Python, https://www.datacamp.com/community/tutorials/discovering-hidden-topics-python

Topic Modeling
Latent Dirichlet Allocation (LDA)



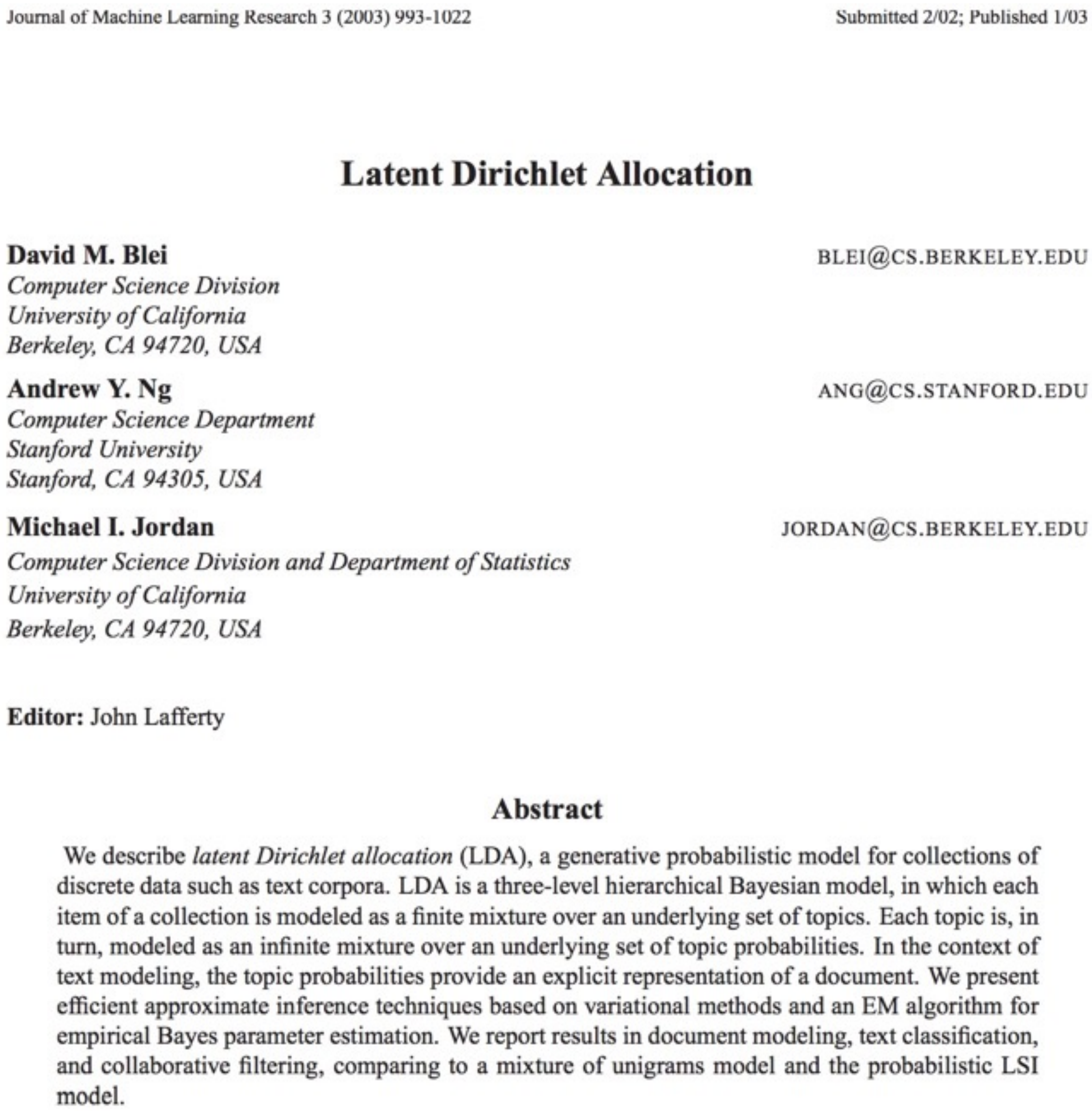
D **documents**

N **words**

K **topics**

Source: Hoffman, Matthew D., David M. Blei, Chong Wang, and John Paisley. "Stochastic variational inference." The Journal of Machine Learning Research 14, no. 1 (2013): 1303-1347.

Latent Dirichlet Allocation (Blei et al., 2003)



Journal of Machine Learning Research 3 (2003) 993-1022 Submitted 2/02; Published 1/03

Latent Dirichlet Allocation

David M. Blei BLEI@CS.BERKELEY.EDU
Computer Science Division
University of California
Berkeley, CA 94720, USA

Andrew Y. Ng ANG@CS.STANFORD.EDU
Computer Science Department
Stanford University
Stanford, CA 94305, USA

Michael I. Jordan JORDAN@CS.BERKELEY.EDU
Computer Science Division and Department of Statistics
University of California
Berkeley, CA 94720, USA

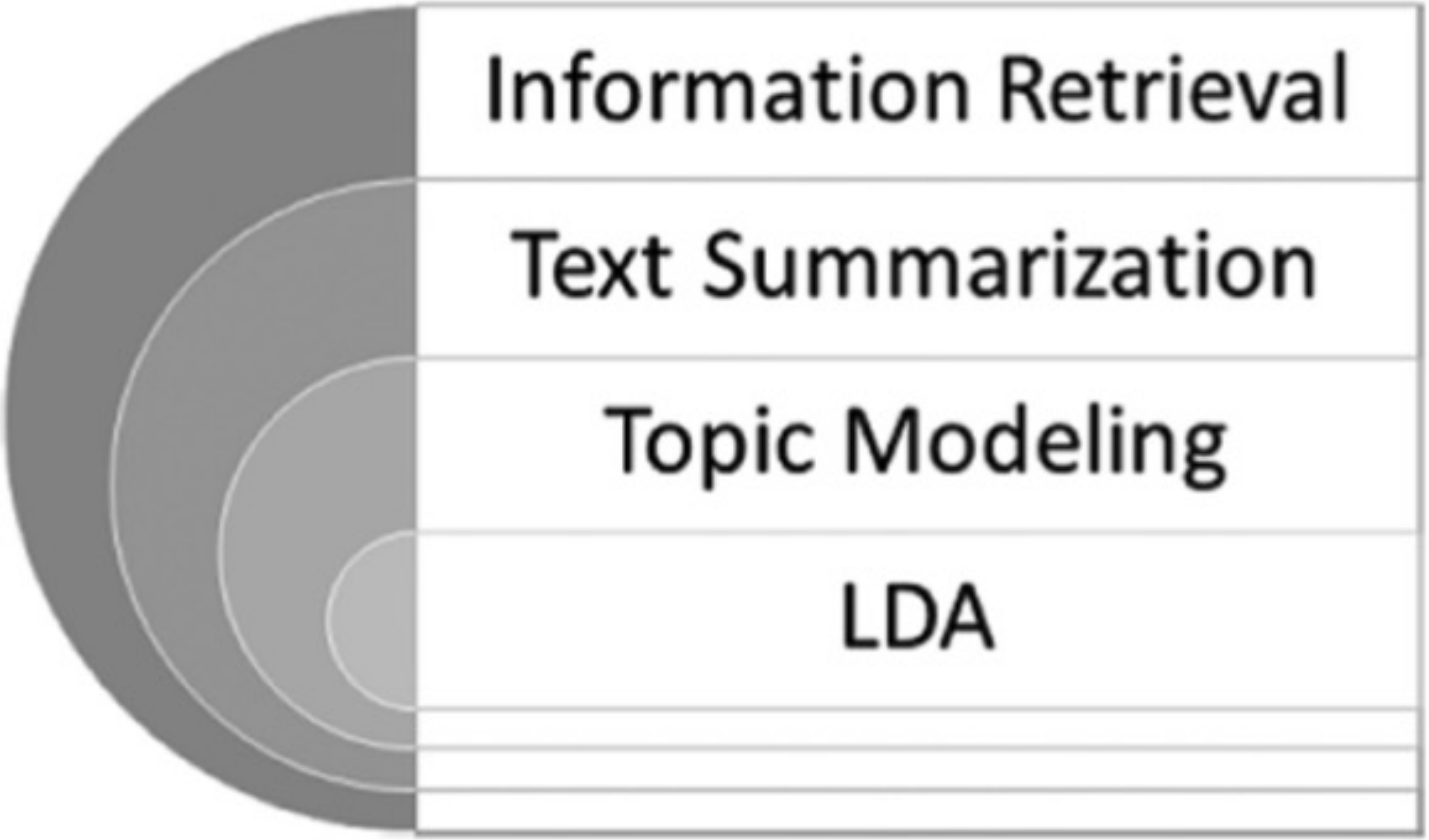
Editor: John Lafferty

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.

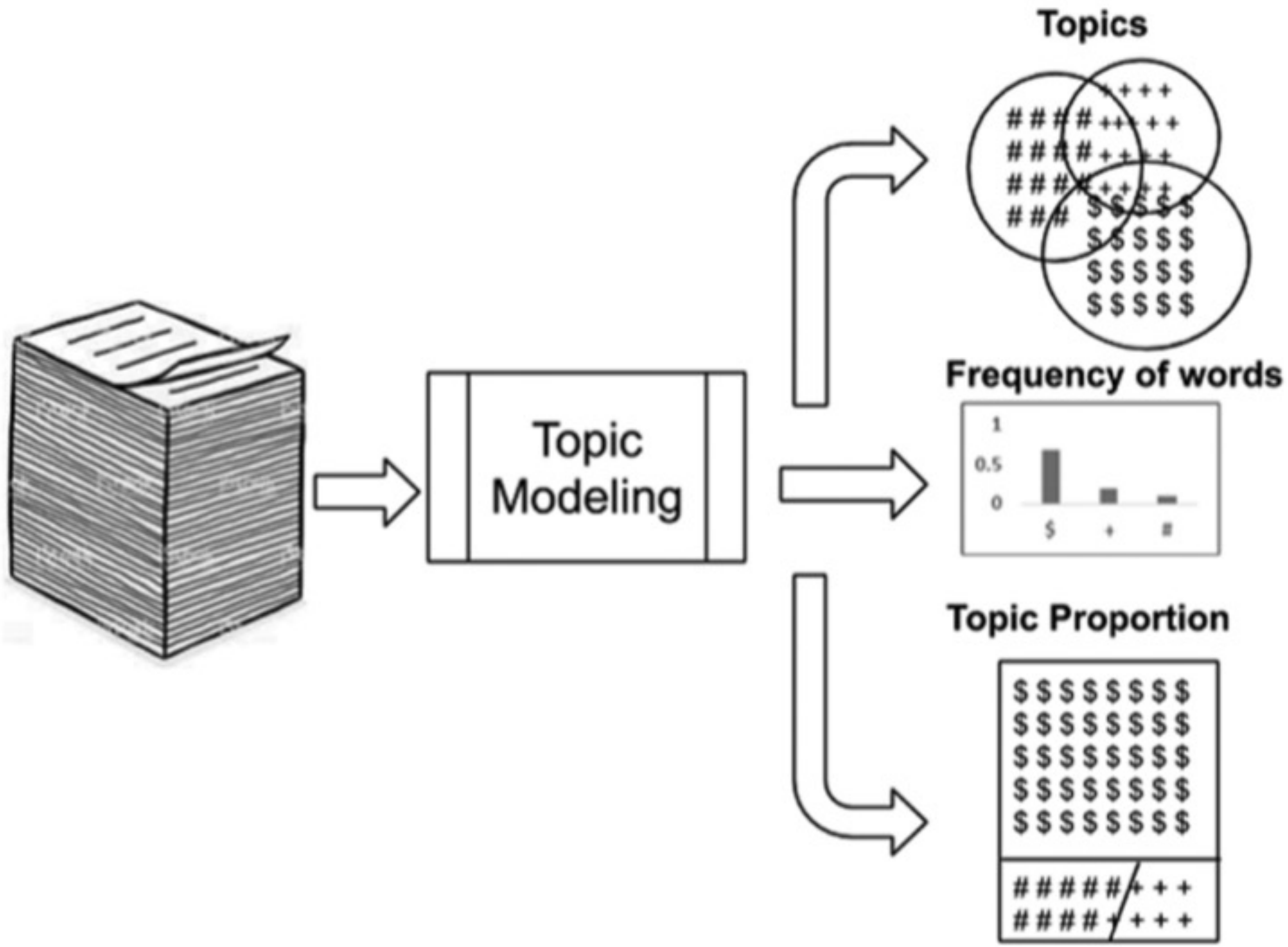
Source: Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." Journal of machine Learning research 3, no. Jan (2003): 993-1022.

Topic Modeling Using Latent Dirichlet allocation (LDA)



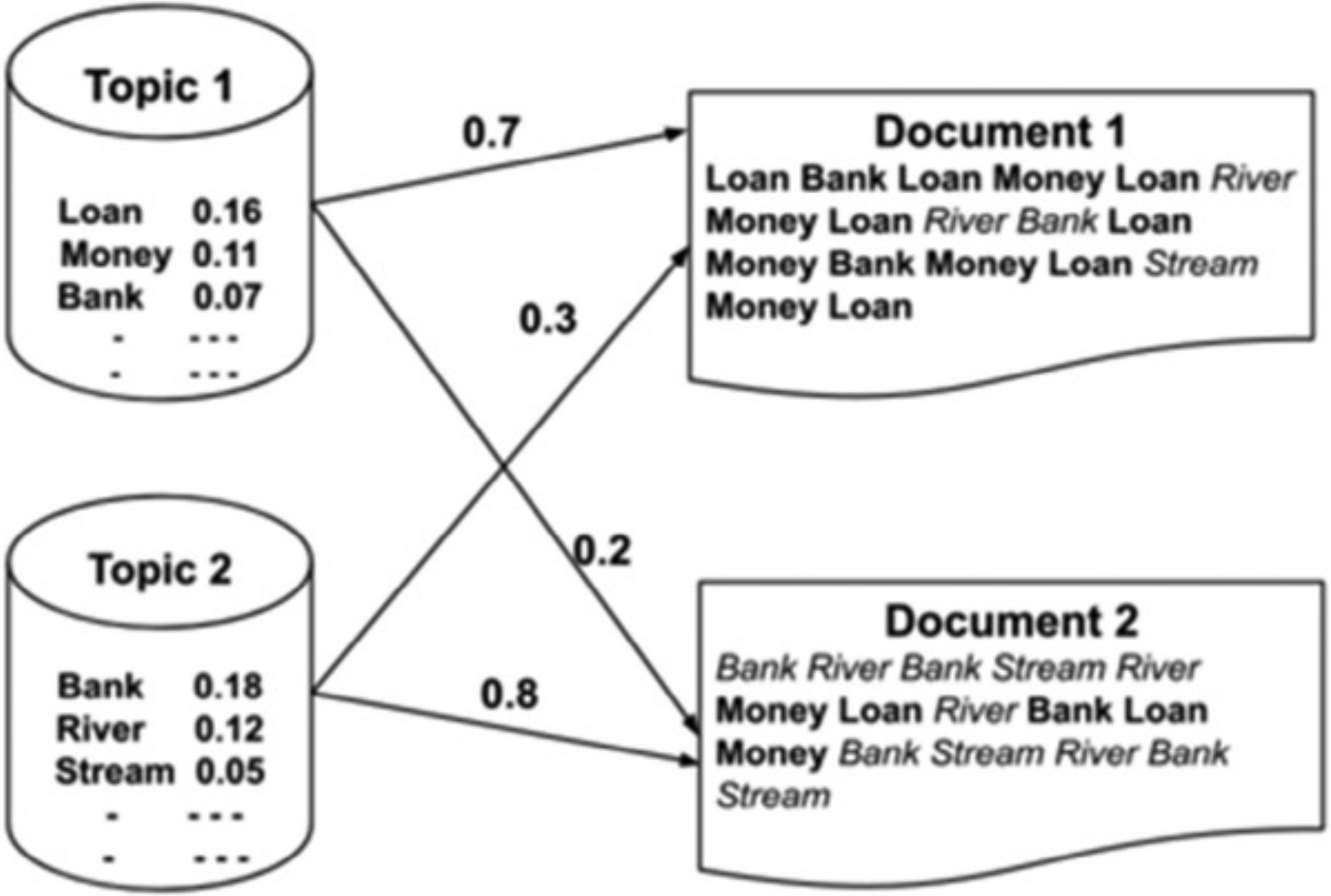
Source: Uttam Chauhan, and Apurva Shah (2021). "Topic modeling using latent Dirichlet allocation: A survey." ACM Computing Surveys (CSUR) 54, no. 7 (2021): 1-35.

Topic Modeling Technique



Source: Uttam Chauhan, and Apurva Shah (2021). "Topic modeling using latent Dirichlet allocation: A survey." ACM Computing Surveys (CSUR) 54, no. 7 (2021): 1-35.

The Generative Process of Latent Dirichlet Allocation (LDA)


Topic 1
Loan 0.16
Money 0.11
Bank 0.07
. ...
. ...
Topic 2
Bank 0.18
River 0.12
Stream 0.05
. ...
. ...
0.7
0.3
0.2
0.8
Document 1
Loan Bank Loan Money Loan *River*
Money Loan *River Bank* Loan
Money Bank Money Loan *Stream*
Money Loan
Document 2
Bank River Bank Stream River
Money Loan *River* Bank Loan
Money *Bank Stream River Bank*
Stream

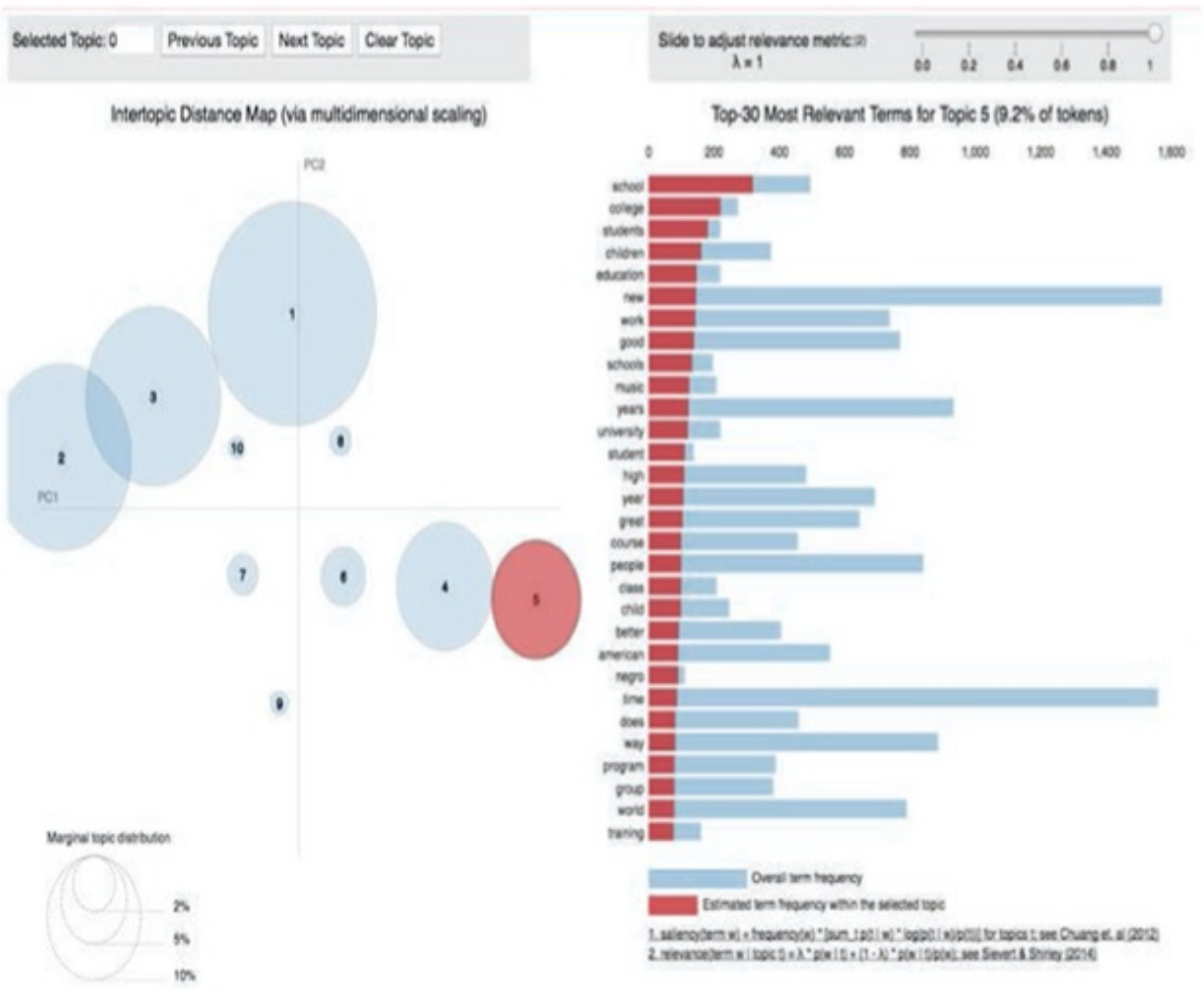
Source: Uttam Chauhan, and Apurva Shah (2021). "Topic modeling using latent Dirichlet allocation: A survey." ACM Computing Surveys (CSUR) 54, no. 7 (2021): 1-35.

Topic Visualization as Word Clouds

Token
Latent
NewWord
random
Topic
proportion
Techniques
Mixture
Textual
Hidden
Models
Reduction
Reassign
document
Multinomial
probability
Allocation
Estimation
Inference

Source: Uttam Chauhan, and Apurva Shah (2021). "Topic modeling using latent Dirichlet allocation: A survey." ACM Computing Surveys (CSUR) 54, no. 7 (2021): 1-35.

LDAvis: Gensim Topic Model Visualization



Selected Topic: 0

Slide to adjust relevance metric:(2)
λ = 1

1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Source: Uttam Chauhan, and Apurva Shah (2021). "Topic modeling using latent Dirichlet allocation: A survey." ACM Computing Surveys (CSUR) 54, no. 7 (2021): 1-35.

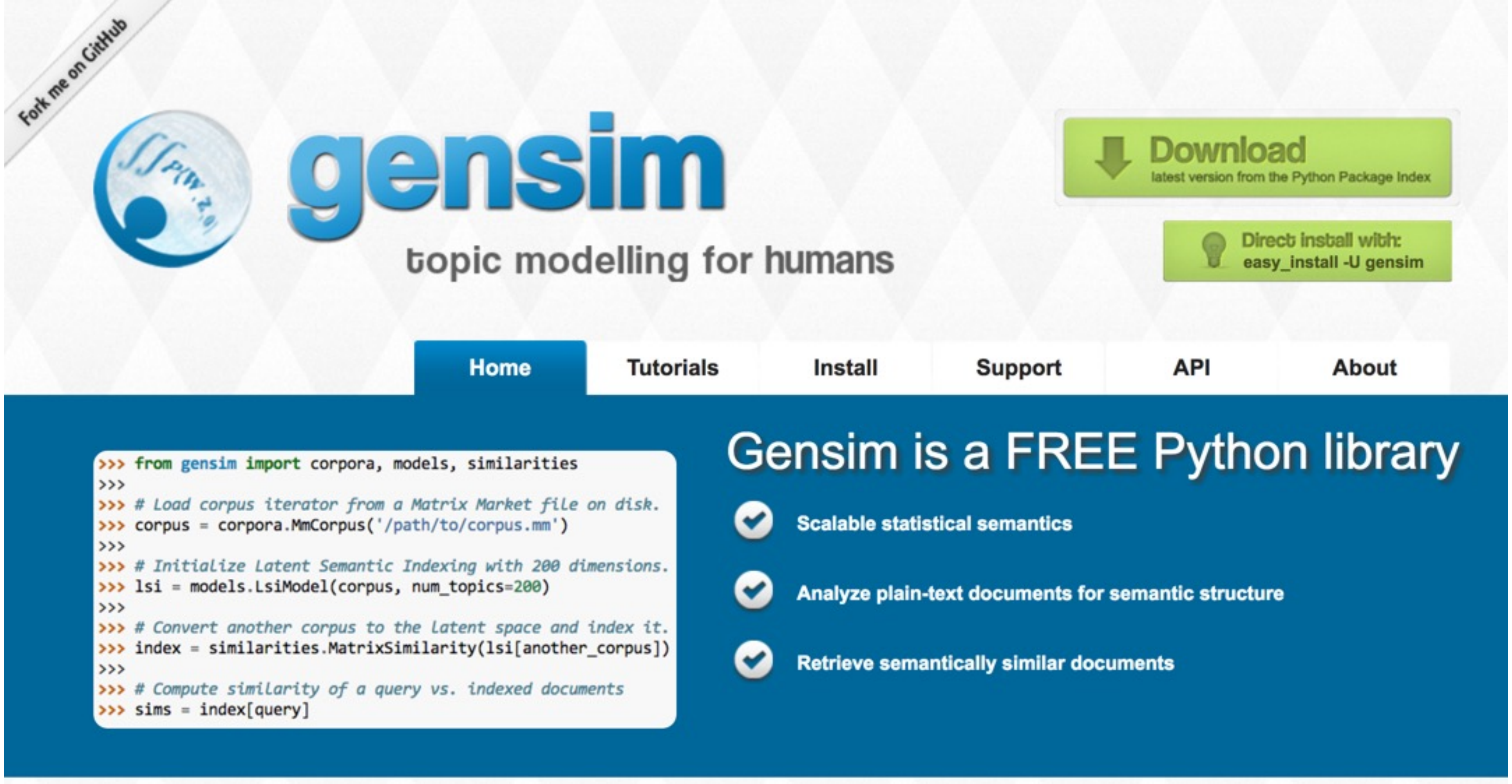
BERTopic

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

Maarten Grootendorst (2022). "BERTopic: Neural topic modeling with a class-based TF-IDF procedure." arXiv preprint arXiv:2203.05794 (2022).

https://github.com/MaartenGr/BERTopic

gensim


Fork me on GitHub
gensim
topic modelling for humans
Download
latest version from the Python Package Index
Home
Tutorials
Install
Support
API
About

Direct install with:
easy_install -U gensim

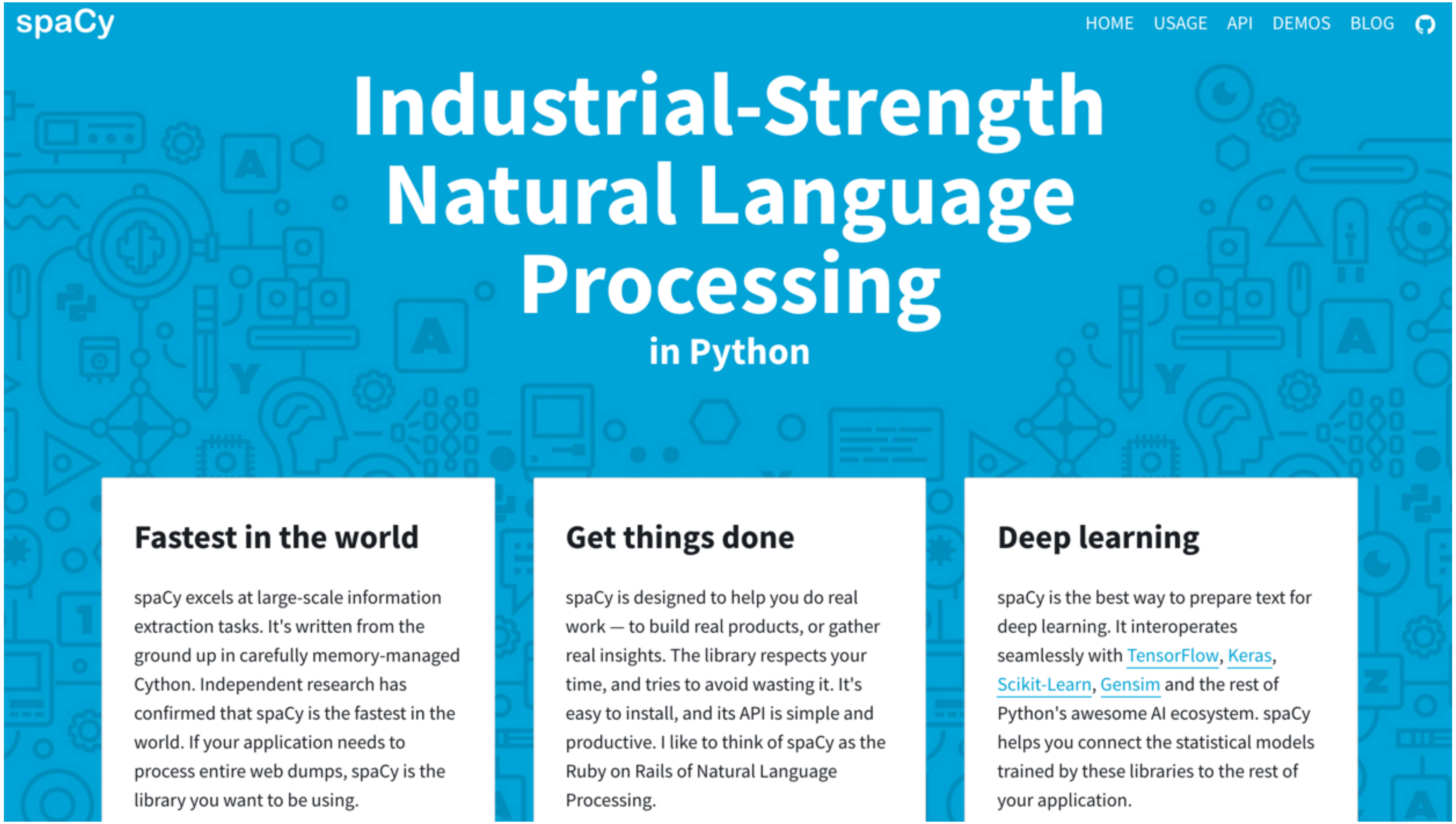
```
>>> from gensim import corpora, models, similarities
>>>
>>> # Load corpus iterator from a Matrix Market file on disk.
>>> corpus = corpora.MmCorpus('/path/to/corpus.mm')
>>>
>>> # Initialize Latent Semantic Indexing with 200 dimensions.
>>> lsi = models.LsiModel(corpus, num_topics=200)
>>>
>>> # 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]
```

Gensim is a FREE Python library

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

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.

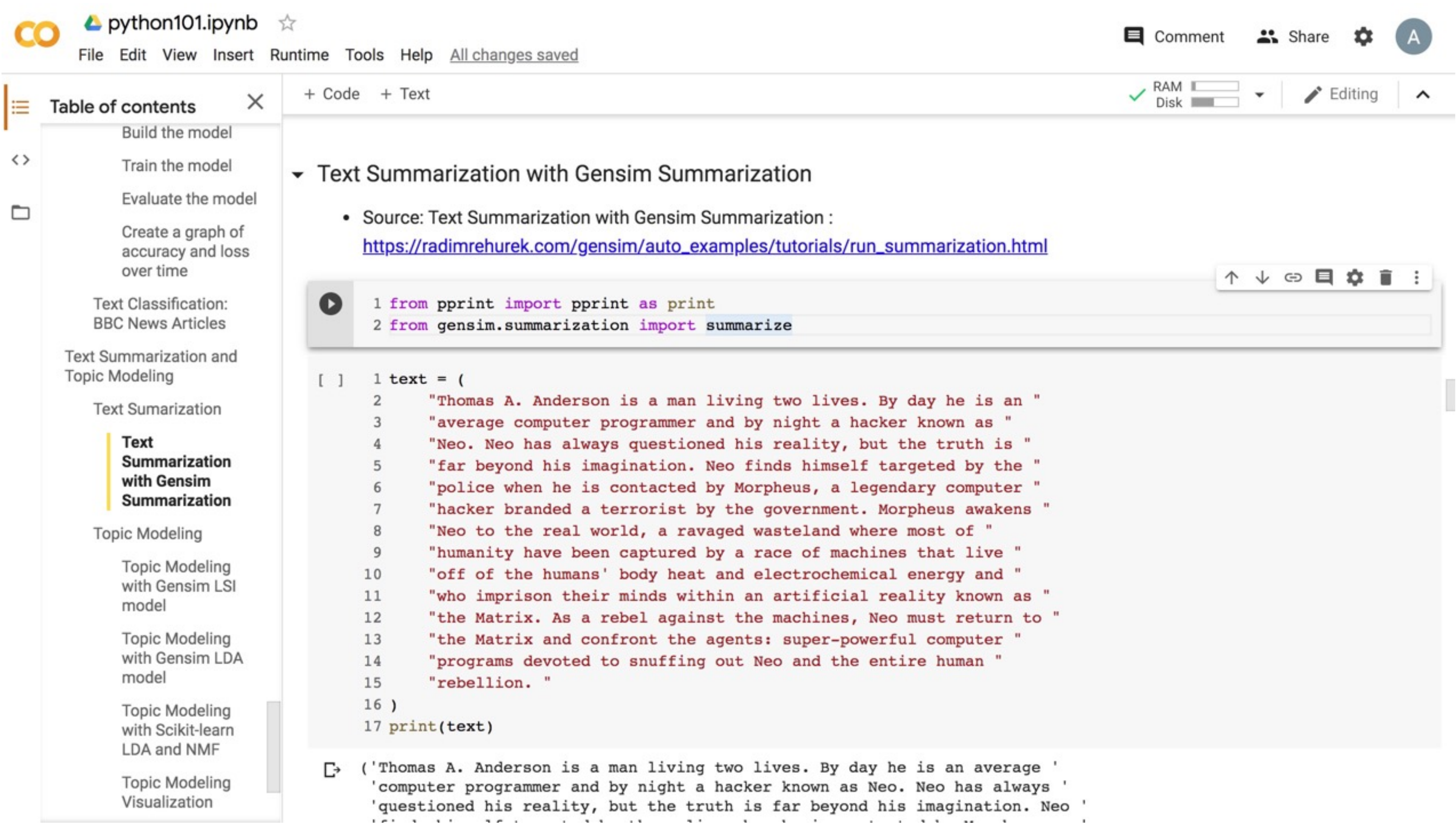
Deep learning

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


python101.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
Table of contents
Build the model
Train the model
Evaluate the model
Create a graph of accuracy and loss over time
Text Classification: BBC News Articles
Text Summarization and Topic Modeling
Text Sumarization
Text Summarization with Gensim Summarization
Topic Modeling
Topic Modeling with Gensim LSI model
Topic Modeling with Gensim LDA model
Topic Modeling with Scikit-learn LDA and NMF
Topic Modeling Visualization

Text Summarization with Gensim Summarization

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

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

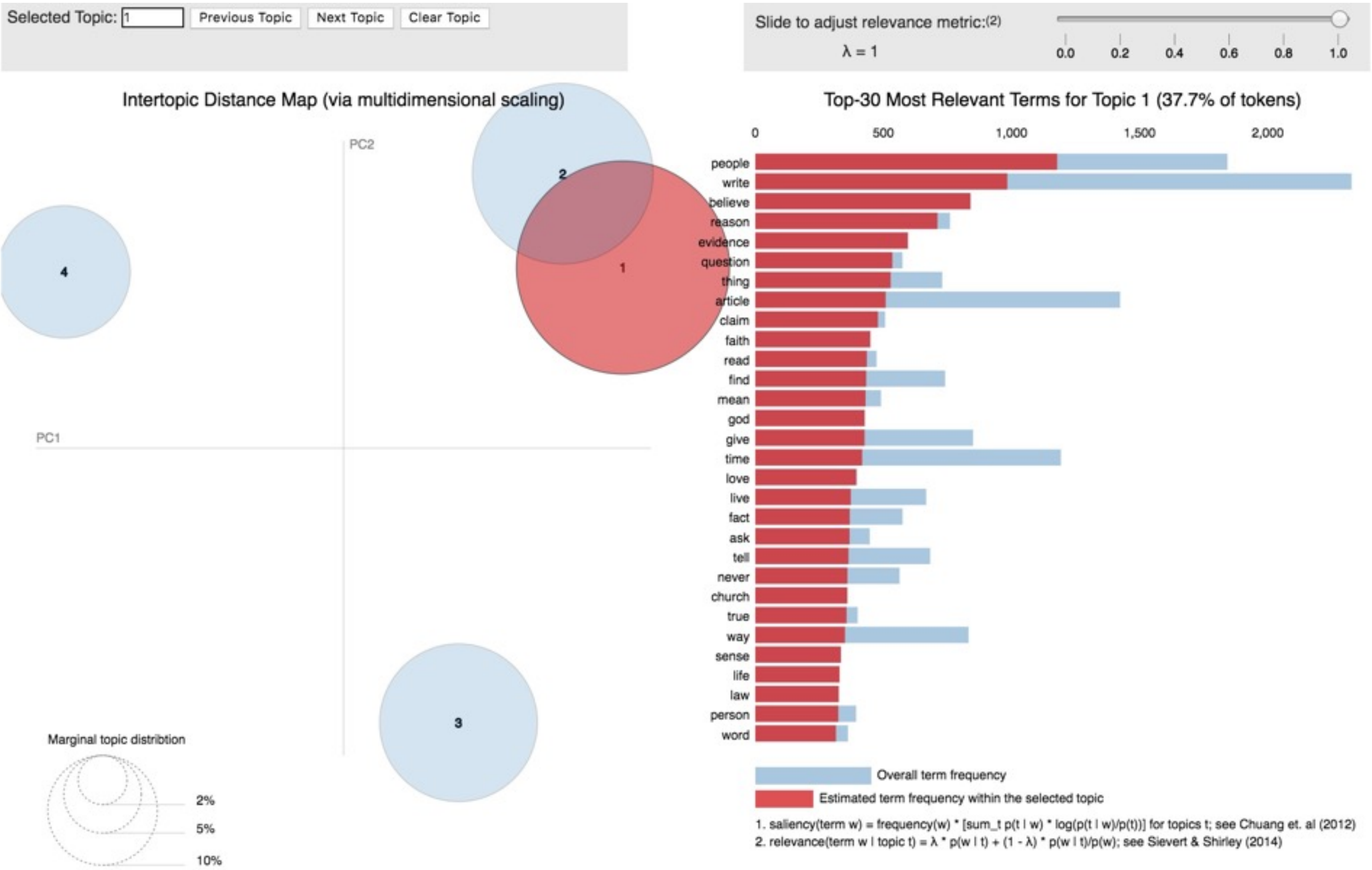
```
text = (
    "Thomas A. Anderson is a man living two lives. By day he is an "
    "average computer programmer and by night a hacker known as "
    "Neo. Neo has always questioned his reality, but the truth is "
    "far beyond his imagination. Neo finds himself targeted by the "
    "police when 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)
```

```
('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 '
```

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

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



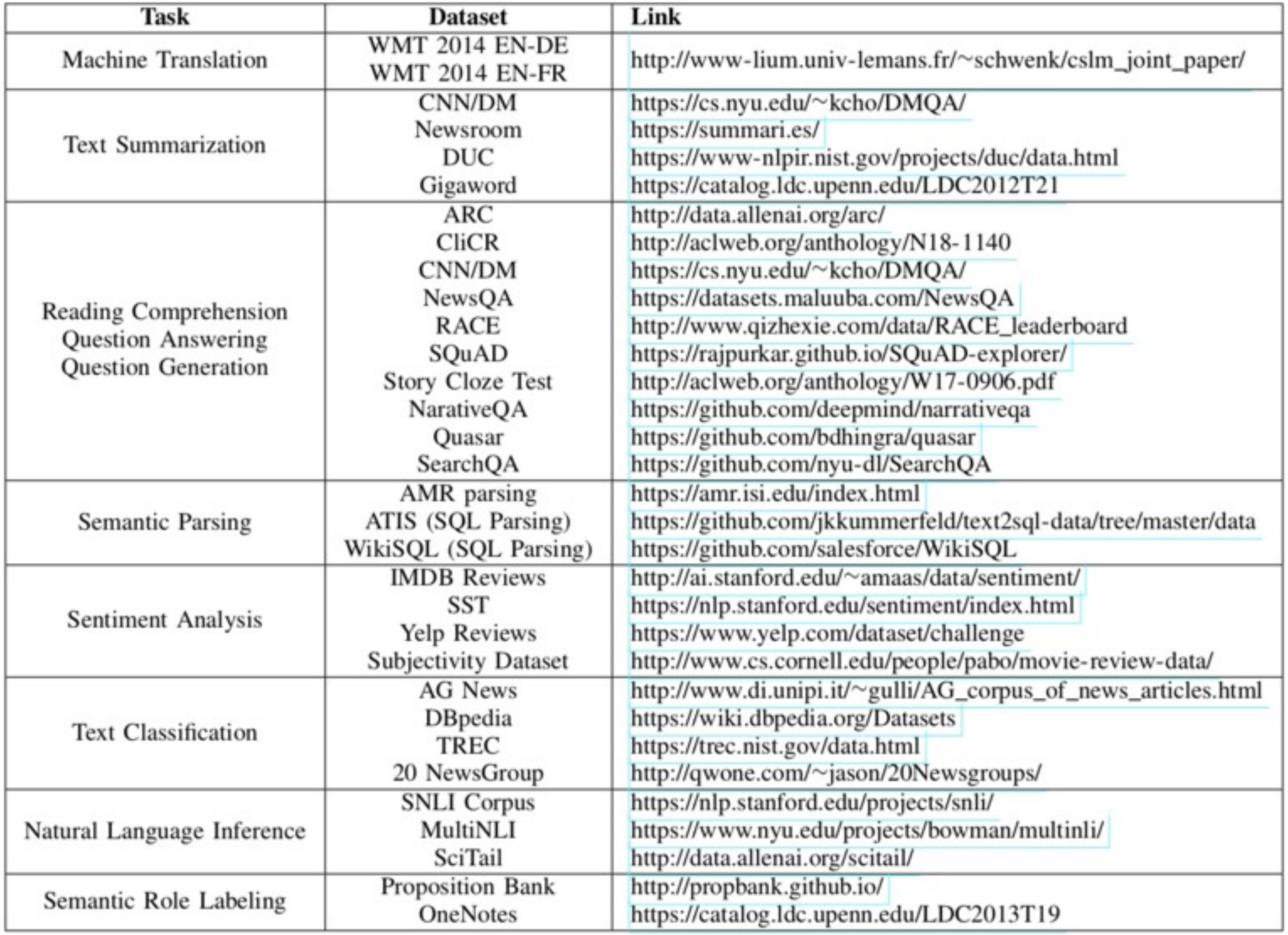
Selected Topic: 1

Slide to adjust relevance metric:(2) λ = 1

1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

https://tinyurl.com/aintpupython101

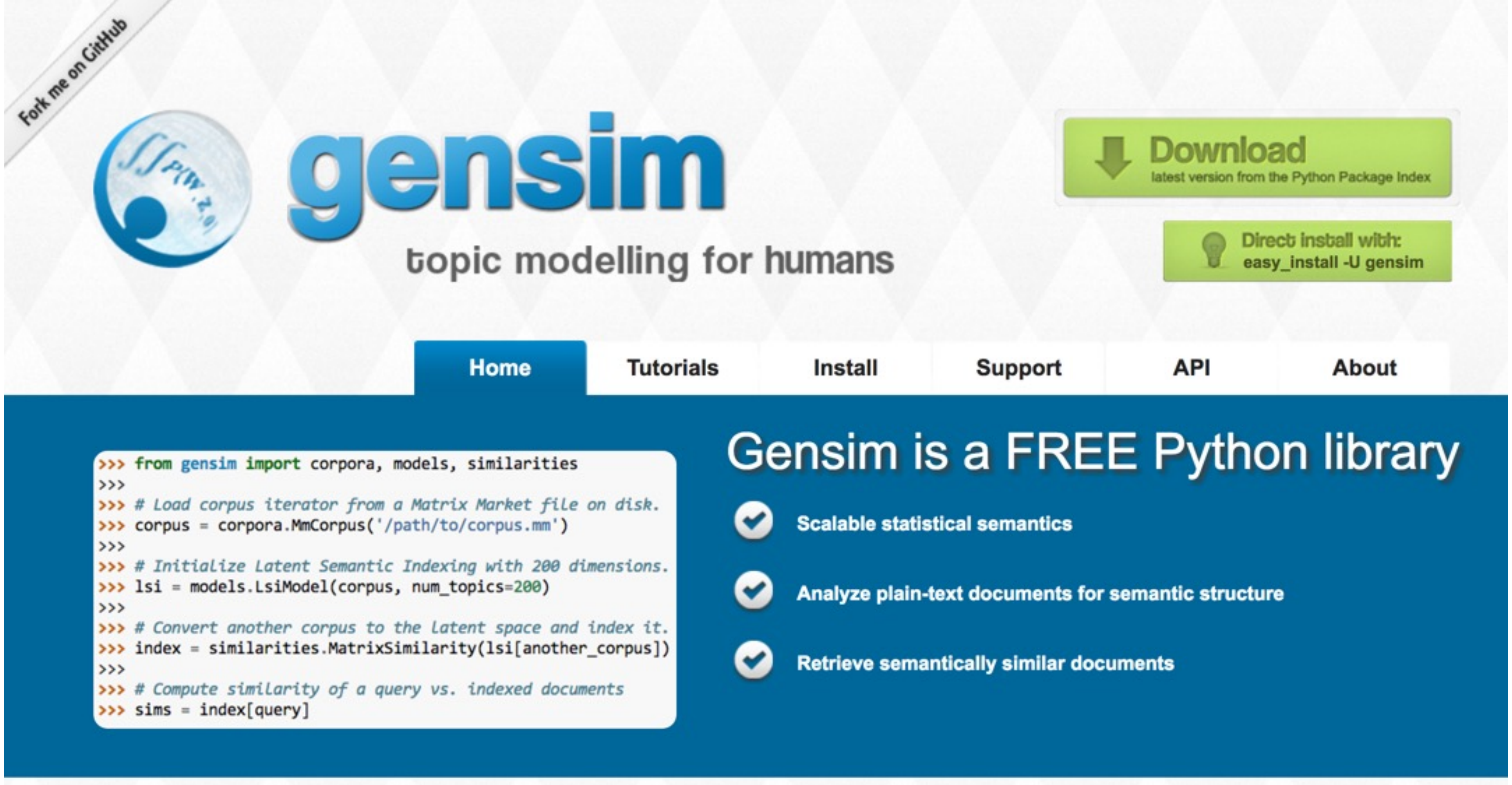
NLP Benchmark Datasets



Task	Dataset	Link
Machine Translation	WMT 2014 EN-DE WMT 2014 EN-FR	http://www-lium.univ-lemans.fr/~schwenk/cslm_joint_paper/
Text Summarization	CNN/DM Newsroom DUC Gigaword	https://cs.nyu.edu/~kcho/DMQA/ https://summari.es/ https://www-nlpir.nist.gov/projects/duc/data.html https://catalog.ldc.upenn.edu/LDC2012T21
Reading Comprehension Question Answering Question Generation	ARC CliCR CNN/DM NewsQA RACE SQuAD Story Cloze Test NarativeQA Quasar SearchQA	http://data.allenai.org/arc/ http://aclweb.org/anthology/N18-1140 https://cs.nyu.edu/~kcho/DMQA/ https://datasets.maluuba.com/NewsQA http://www.qizhexie.com/data/RACE_leaderboard https://rajpurkar.github.io/SQuAD-explorer/ http://aclweb.org/anthology/W17-0906.pdf https://github.com/deepmind/narrativeqa https://github.com/bdhingra/quasar https://github.com/nyu-dl/SearchQA
Semantic Parsing	AMR parsing ATIS (SQL Parsing) WikiSQL (SQL Parsing)	https://amr.isi.edu/index.html https://github.com/jkkummerfeld/text2sql-data/tree/master/data https://github.com/salesforce/WikiSQL
Sentiment Analysis	IMDB Reviews SST Yelp Reviews Subjectivity Dataset	http://ai.stanford.edu/~amaas/data/sentiment/ https://nlp.stanford.edu/sentiment/index.html https://www.yelp.com/dataset/challenge http://www.cs.cornell.edu/people/pabo/movie-review-data/
Text Classification	AG News DBpedia TREC 20 NewsGroup	http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html https://wiki.dbpedia.org/Datasets https://trec.nist.gov/data.html http://qwone.com/~jason/20Newsgroups/
Natural Language Inference	SNLI Corpus MultiNLI SciTail	https://nlp.stanford.edu/projects/snli/ https://www.nyu.edu/projects/bowman/multinli/ http://data.allenai.org/scitail/
Semantic Role Labeling	Proposition Bank OneNotes	http://propbank.github.io/ https://catalog.ldc.upenn.edu/LDC2013T19

Source: Amirsina Torfi, Rouzbeh A. Shirvani, Yaser Keneshloo, Nader Tavvaf, and Edward A. Fox (2020).
"Natural Language Processing Advancements By Deep Learning: A Survey." arXiv preprint arXiv:2003.01200.

gensim



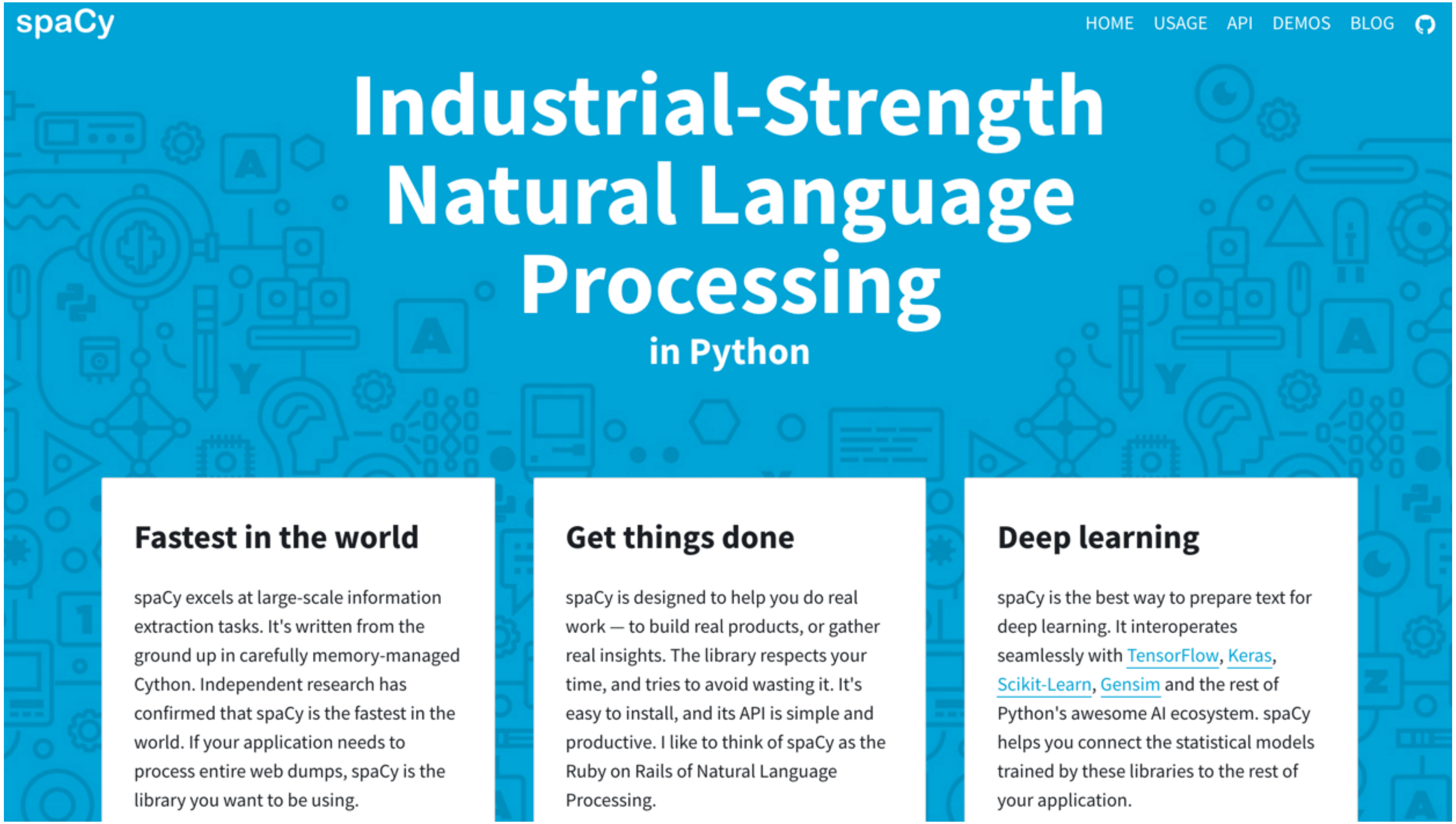
```
>>> from gensim import corpora, models, similarities
>>>
>>> # Load corpus iterator from a Matrix Market file on disk.
>>> corpus = corpora.MmCorpus('/path/to/corpus.mm')
>>>
>>> # Initialize Latent Semantic Indexing with 200 dimensions.
>>> lsi = models.LsiModel(corpus, num_topics=200)
>>>
>>> # 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]
```

Gensim is a FREE Python library

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

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

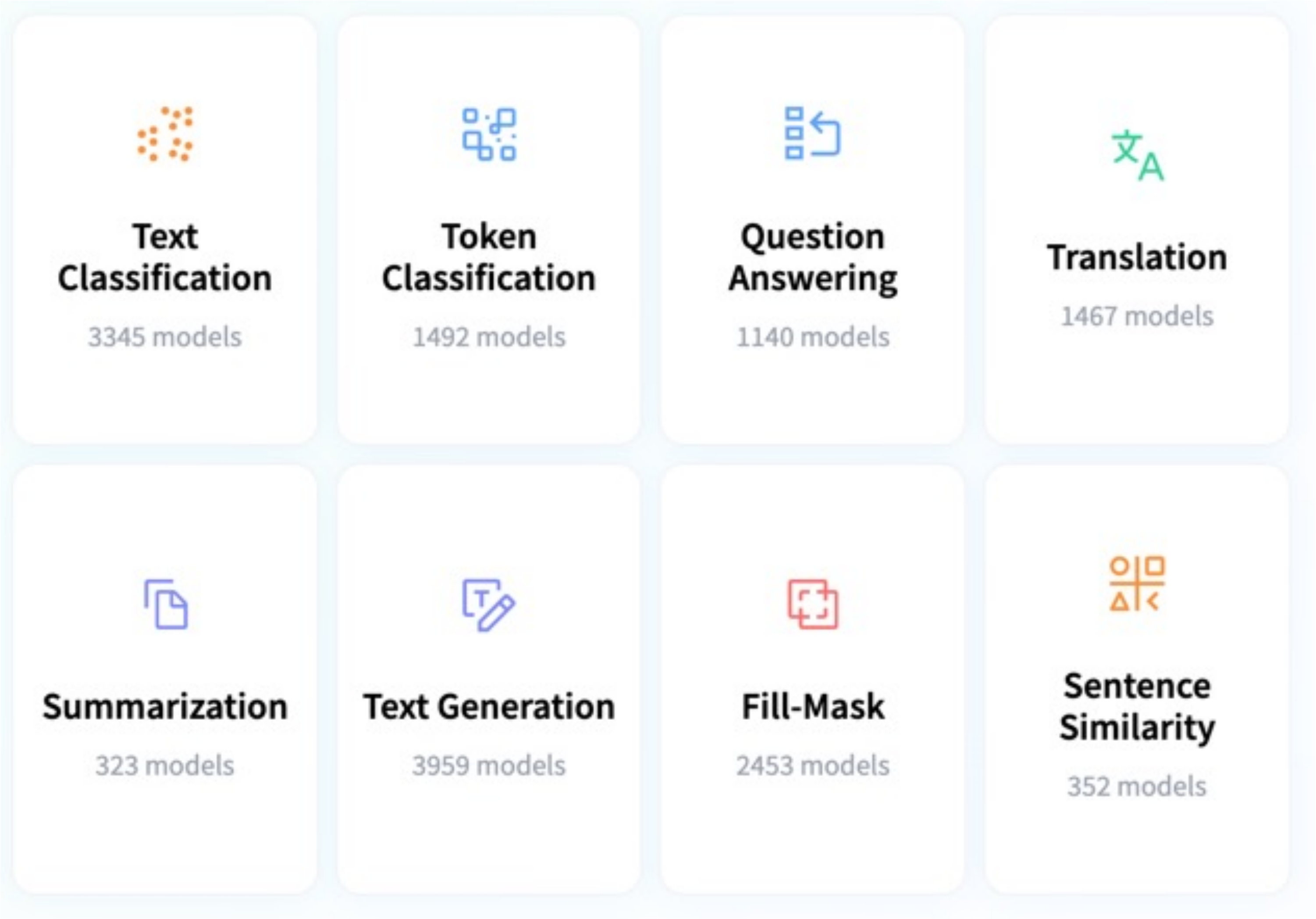
spaCy is designed to help you do real work — to build real products, or gather real insights. The library respects your time, and tries to avoid wasting it. It's easy to install, and its API is simple and productive. I like to think of spaCy as the Ruby on Rails of Natural Language Processing.

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

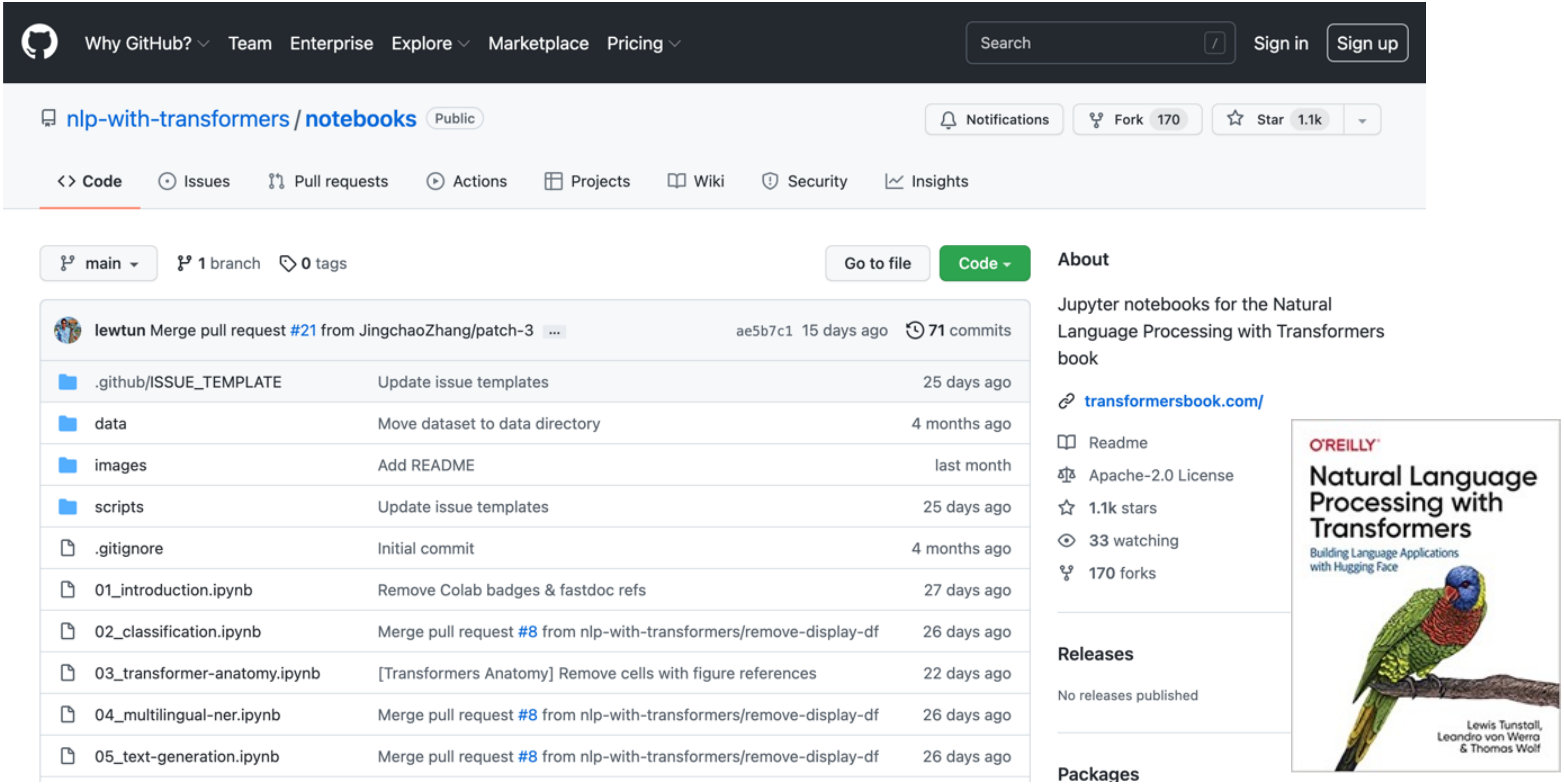
Text Generation
3959 models

Fill-Mask
2453 models

Sentence Similarity
352 models

https://huggingface.co/tasks

NLP with Transformers Github



nlp-with-transformers / notebooks Public

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

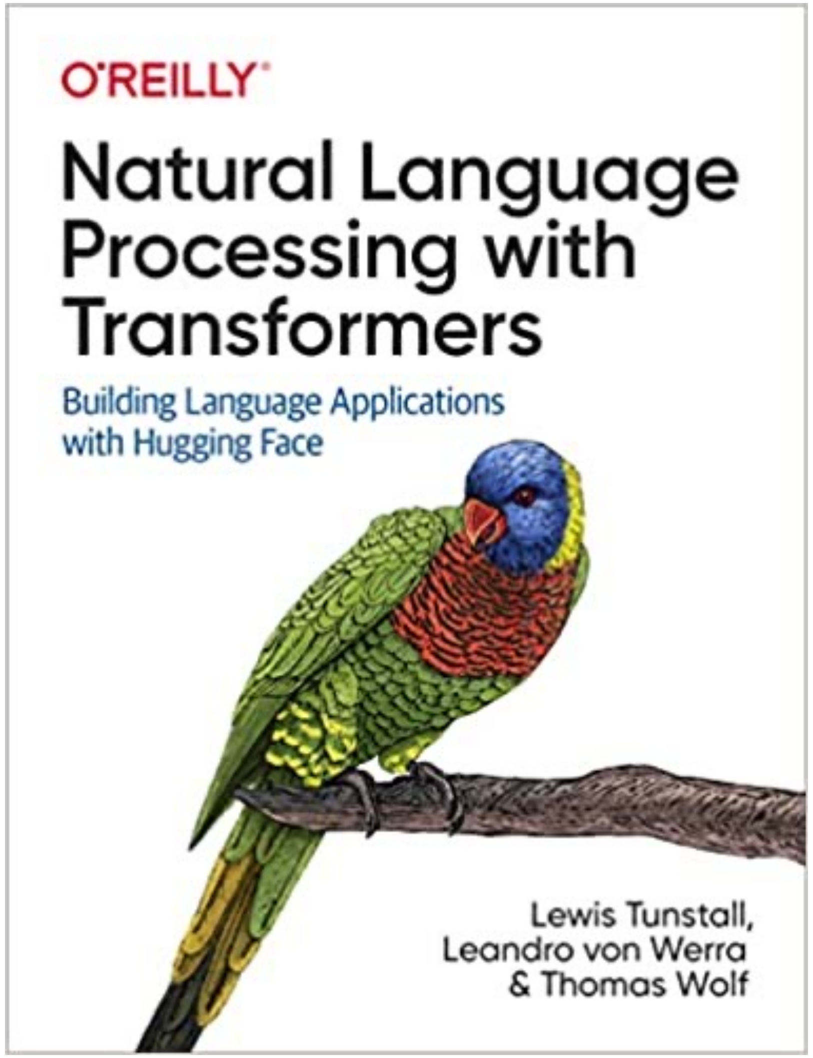
No releases published

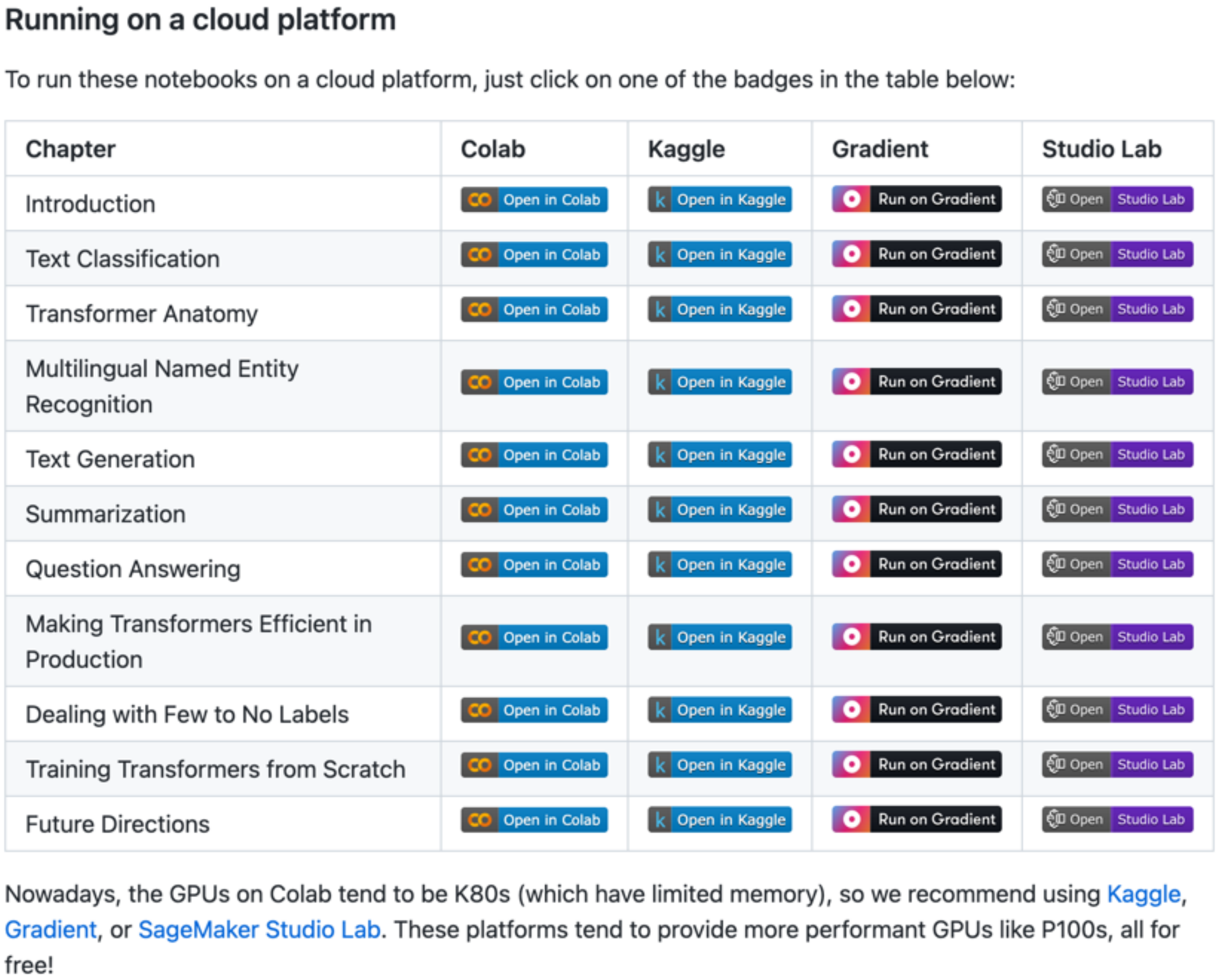
Packages

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:

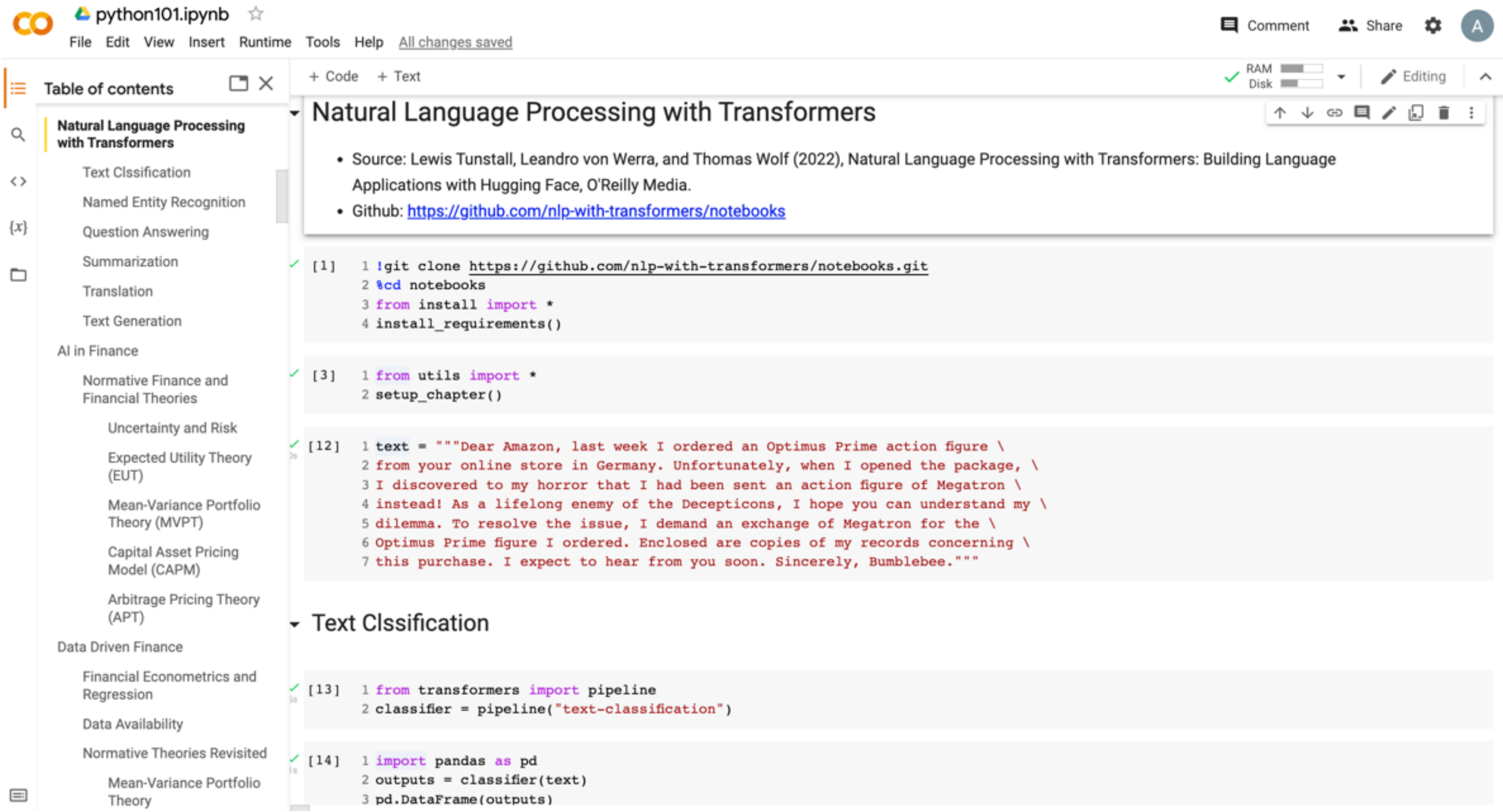
Chapter	Colab	Kaggle	Gradient	Studio Lab
Introduction	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Text Classification	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Transformer Anatomy	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Multilingual Named Entity 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



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

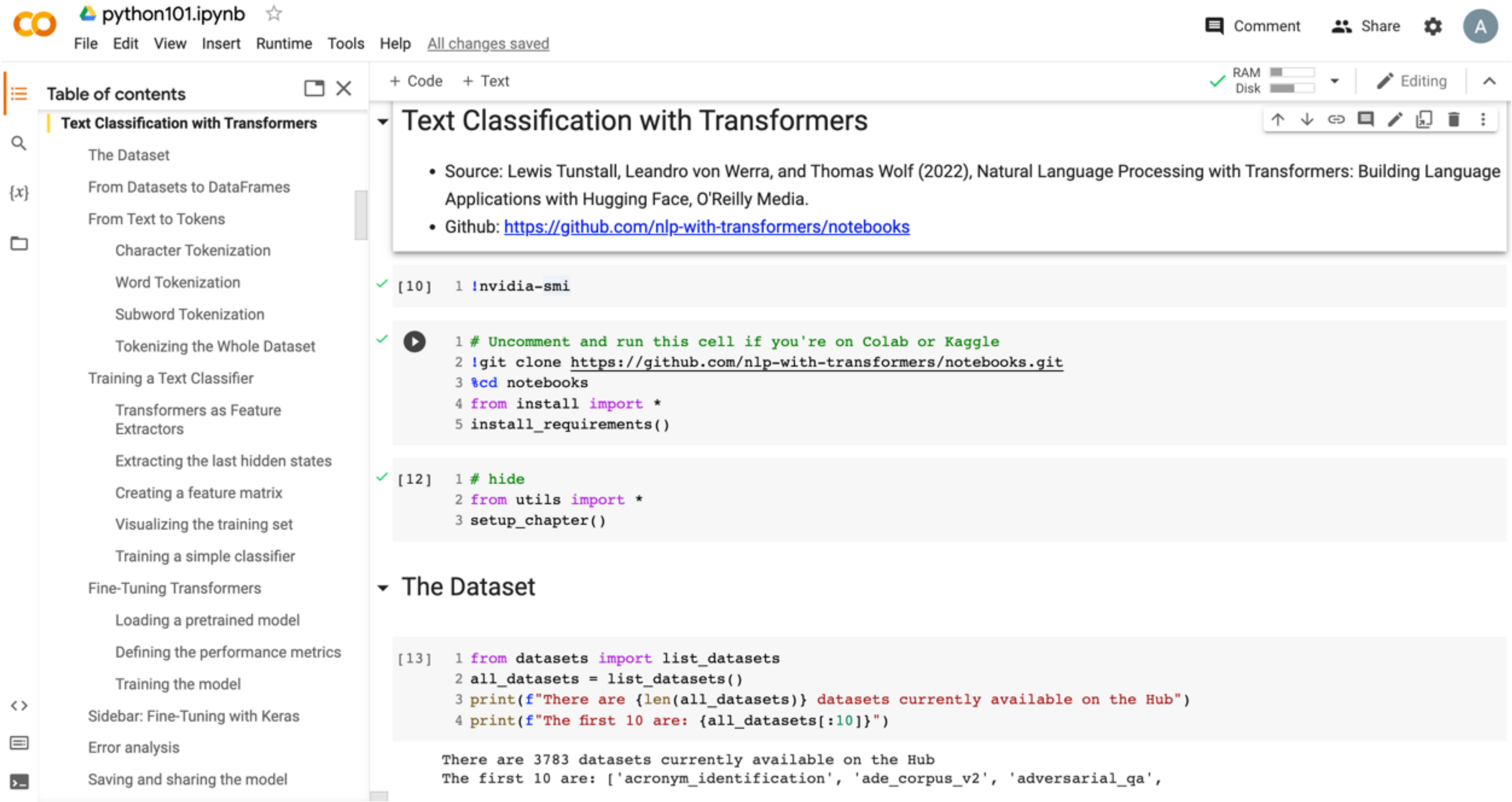


Table of contents

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

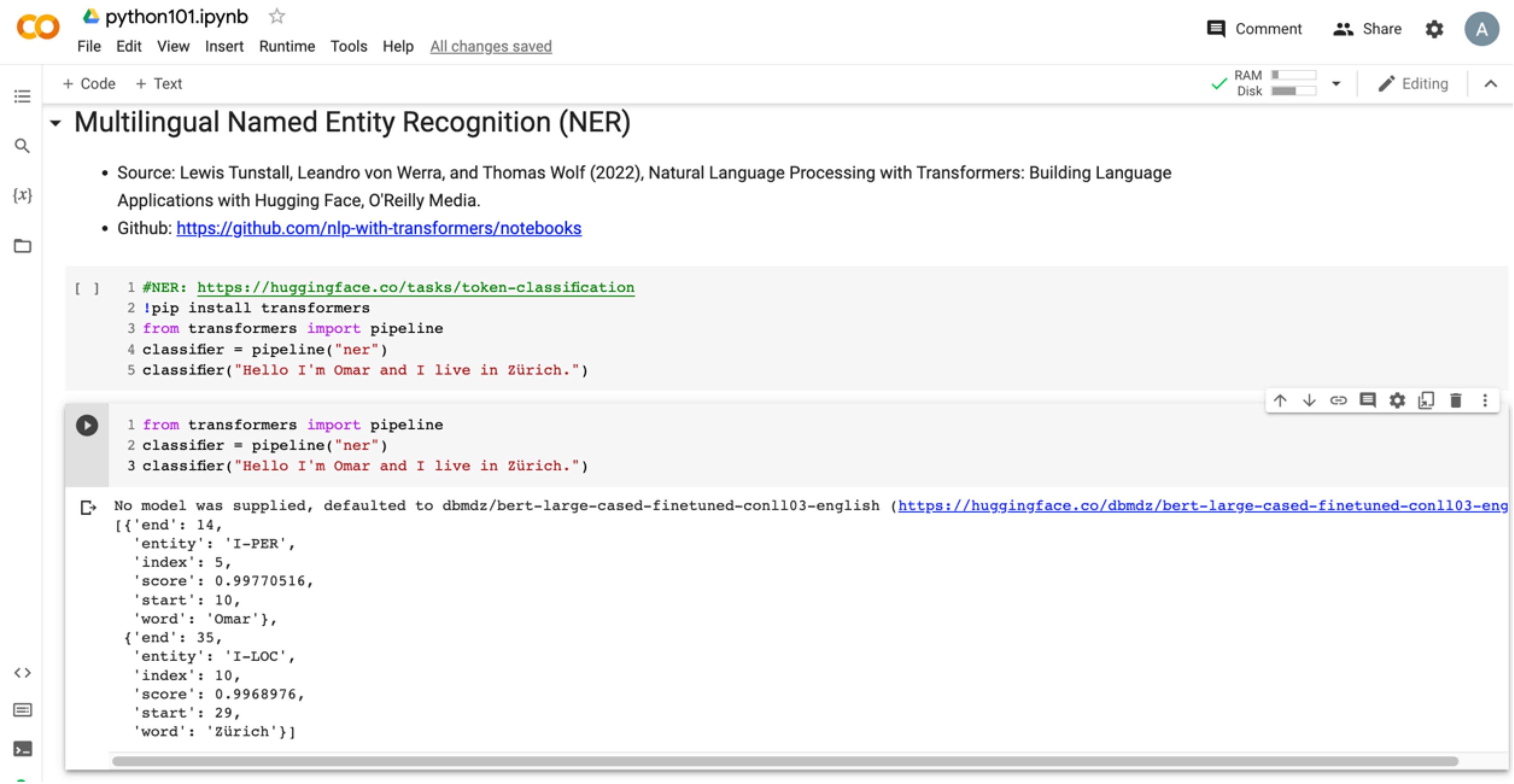
[13]
```
from datasets import list_datasets
all_datasets = list_datasets()
print(f"There are {len(all_datasets)} datasets currently available on the Hub")
print(f"The first 10 are: {all_datasets[:10]}")
```

```
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



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.")
```

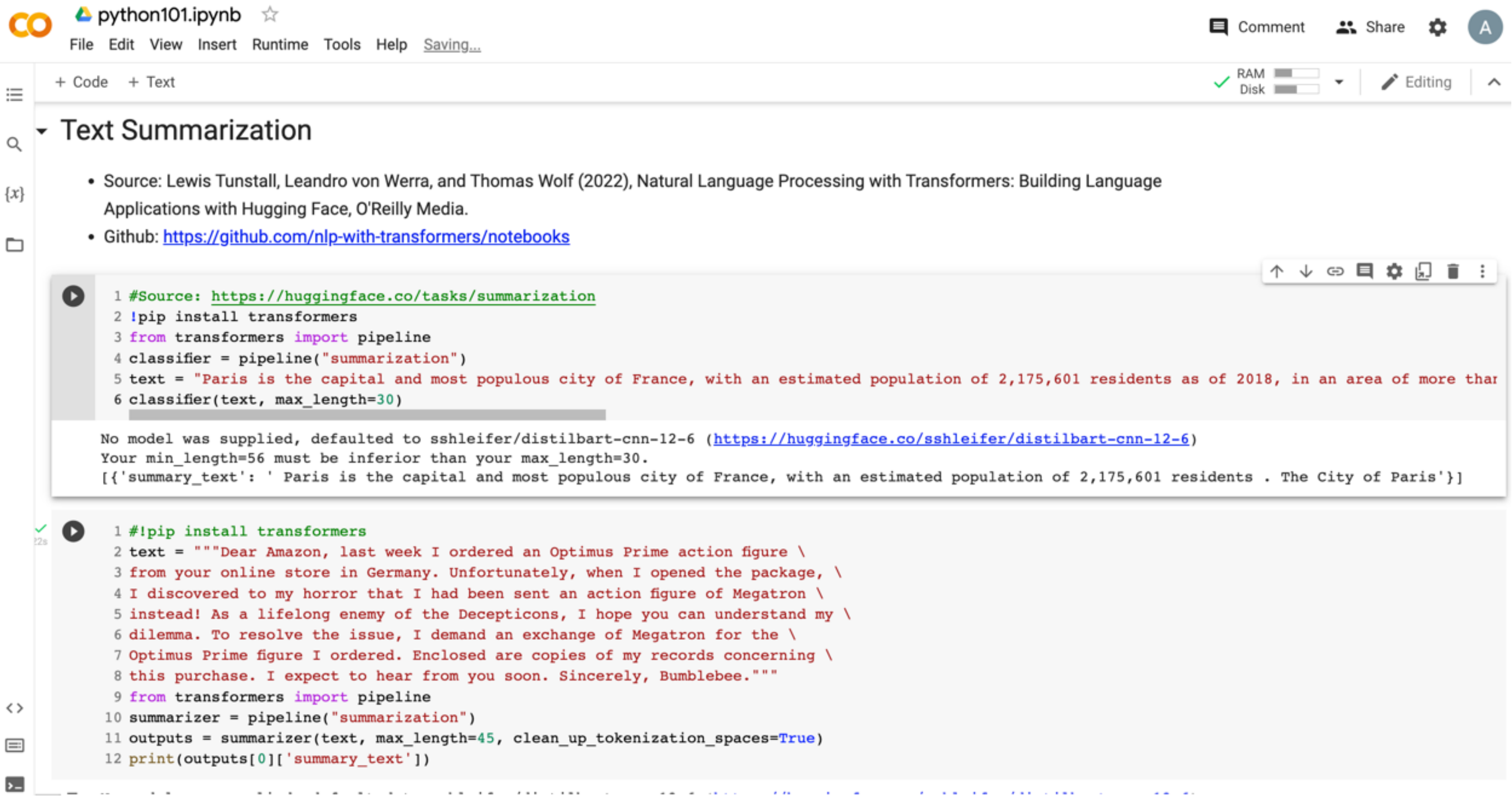
```
from transformers import pipeline
classifier = pipeline("ner")
classifier("Hello I'm Omar and I live in Zürich.")
```

```
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

- 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

Text Summarization

```
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."""
```

```python
from transformers import pipeline
summarizer = pipeline("summarization")
outputs = summarizer(text, max_length=45, clean_up_tokenization_spaces=True)
print(outputs[0]['summary_text'])
```

```
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.
```

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.
- 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.
- Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed (2021). "Automatic text summarization: A comprehensive survey." Expert Systems with Applications 165 (2021): 113679.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu (2019). "Exploring the limits of transfer learning with a unified text-to-text transformer." arXiv preprint arXiv:1910.10683 (2019).
- Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu (2020). "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization." In International Conference on Machine Learning, pp. 11328-11339. PMLR, 2020.
- Uttam Chauhan, and Apurva Shah (2021). "Topic modeling using latent Dirichlet allocation: A survey." ACM Computing Surveys (CSUR) 54, no. 7 (2021): 1-35.
- Maarten Grootendorst (2022). "BERTopic: Neural topic modeling with a class-based TF-IDF procedure." arXiv preprint arXiv:2203.05794 (2022).
- Benjamin Bengfort, Rebecca Bilbro, and Tony Ojeda (2018), Applied Text Analysis with Python: Enabling Language-Aware Data Products with Machine Learning, O'Reilly.
- 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.
- 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