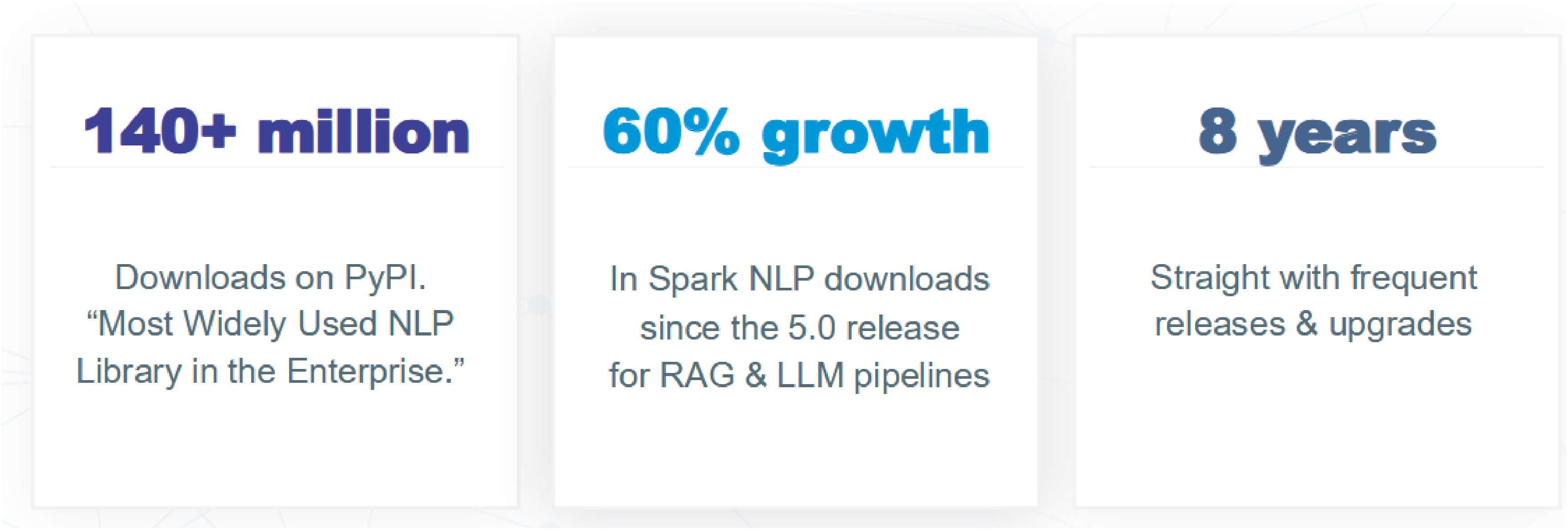
John Snow LABS

Open Source Capabilities

Gursev Pirge
08-04-2025

Open Source Library



140+ million

Downloads on PyPI.
“Most Widely Used NLP
Library in the Enterprise.”

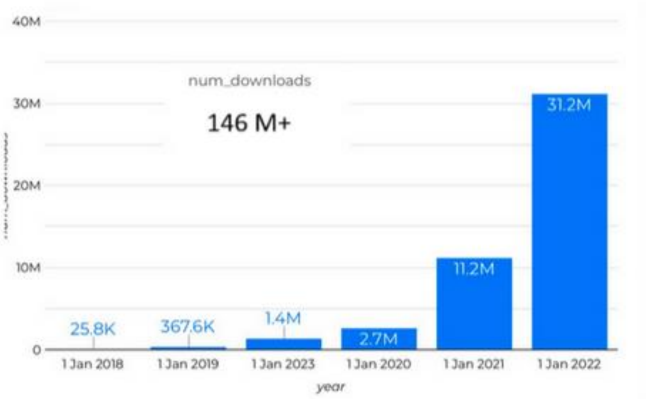
60% growth

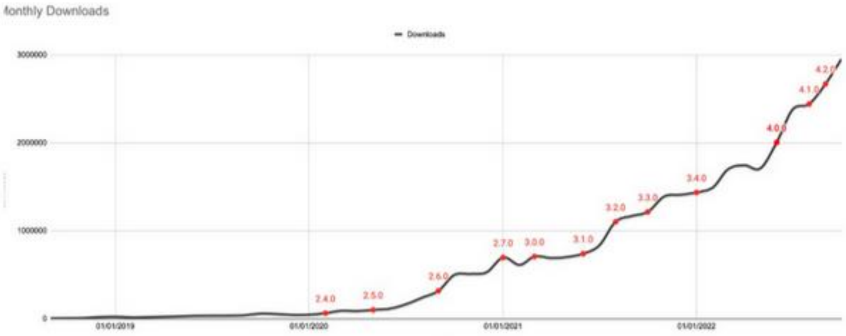
In Spark NLP downloads
since the 5.0 release
for RAG & LLM pipelines

8 years

Straight with frequent
releases & upgrades

Introducing Spark NLP

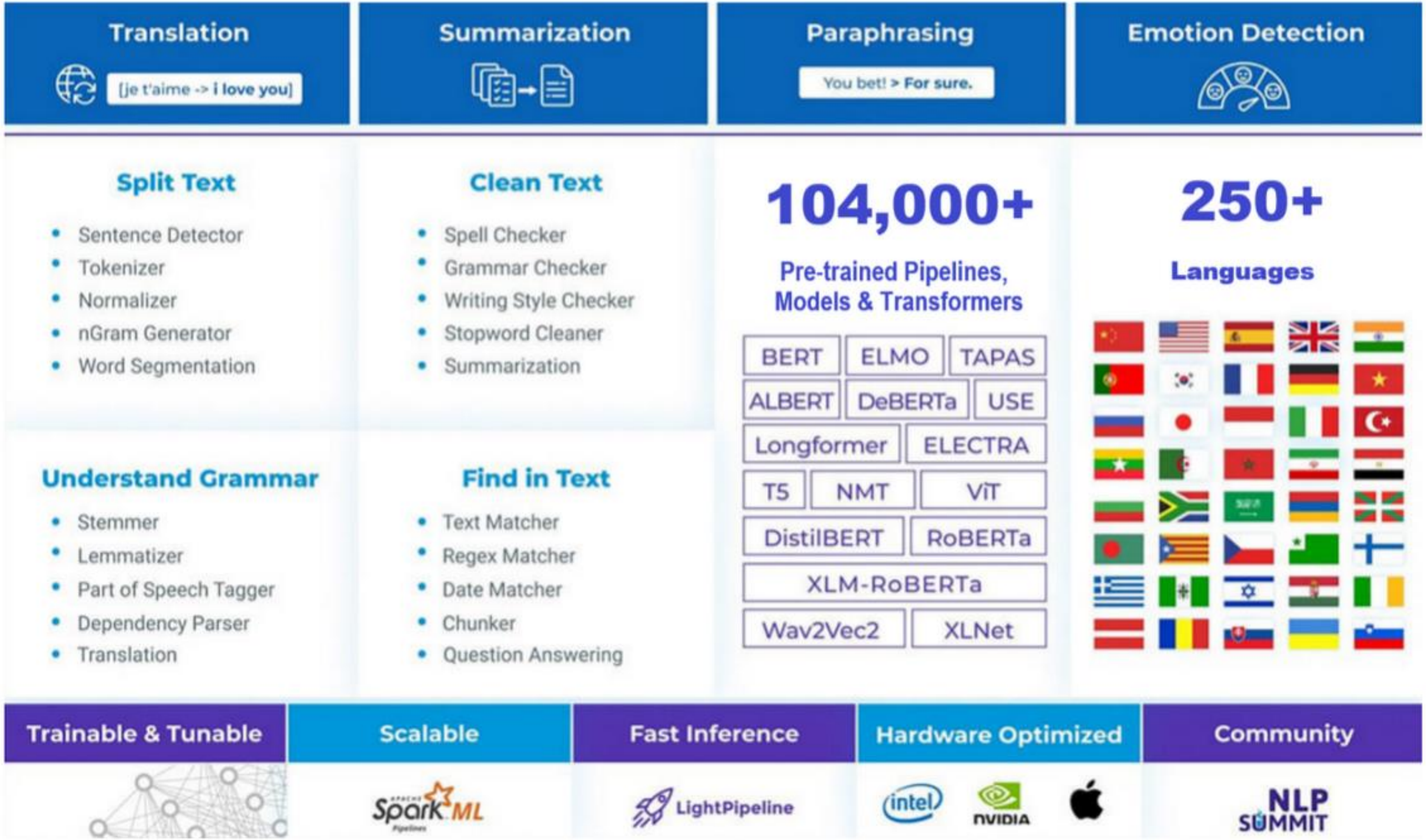




Spark NLP is an open-source natural language processing library, built on top of **Apache Spark** and **Spark ML**. (first release: July 2017)

- A single unified solution for all your NLP needs
- Take advantage of transfer learning and implementing the latest and greatest **SOTA** algorithms and models in NLP research
- The most widely used NLP library in industry (5 yrs in a row)
- The most scalable, accurate and fastest library in NLP history
- 111 total releases, every two weeks for the past 5 years

John Snow LABS



Translation

[je t'aime -> i love you]

Summarization

Paraphrasing

You bet! > For sure.

Emotion Detection

Split Text

- Sentence Detector
- Tokenizer
- Normalizer
- nGram Generator
- Word Segmentation

Clean Text

- Spell Checker
- Grammar Checker
- Writing Style Checker
- Stopword Cleaner
- Summarization

Understand Grammar

- Stemmer
- Lemmatizer
- Part of Speech Tagger
- Dependency Parser
- Translation

Find in Text

- Text Matcher
- Regex Matcher
- Date Matcher
- Chunker
- Question Answering

104,000+

Pre-trained Pipelines, Models & Transformers

BERT | ELMO | TAPAS
ALBERT | DeBERTa | USE
Longformer | ELECTRA
T5 | NMT | ViT
DistilBERT | RoBERTa
XLM-RoBERTa
Wav2Vec2 | XLNet

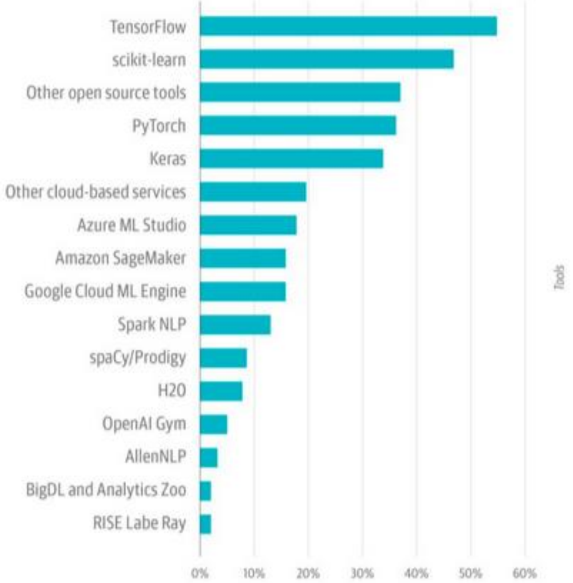
250+

Languages

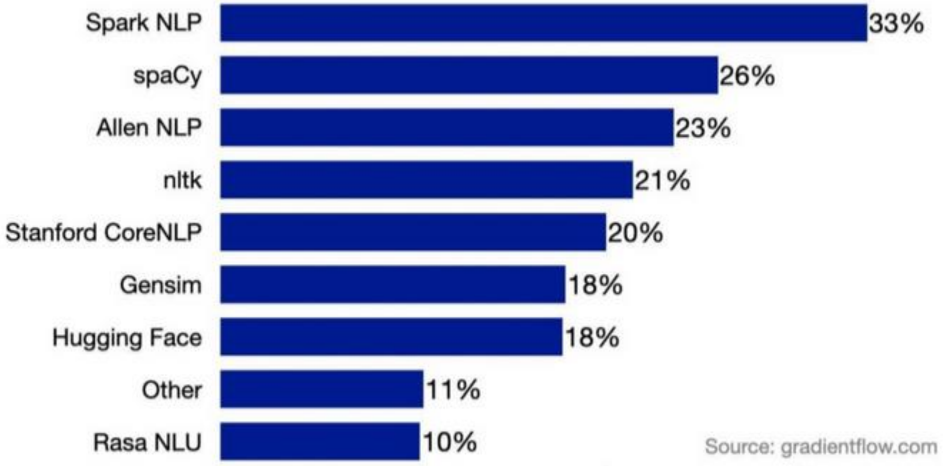
Trainable & Tunable	Scalable	Fast Inference	Hardware Optimized	Community
	Spark ML	LightPipeline	intel NVIDIA	NLP SUMMIT

Spark NLP in Industry

Which of the following AI tools do you use?



Which NLP libraries does your organization use?



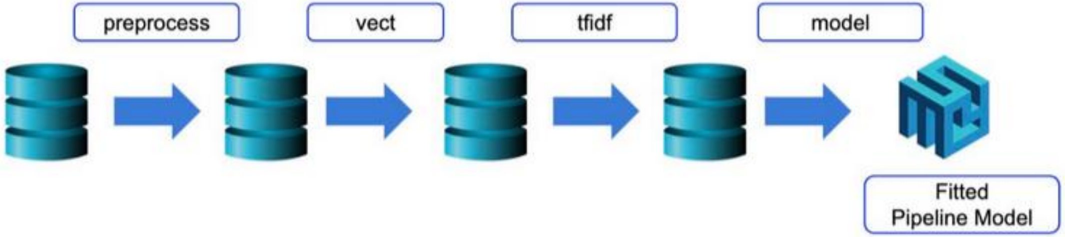
Source: gradientflow.com

NLP Industry Survey by Gradient Flow,
an independent data science research & insights company, September 2021

John Snow LABS

Introducing Spark NLP

Pipeline of annotators



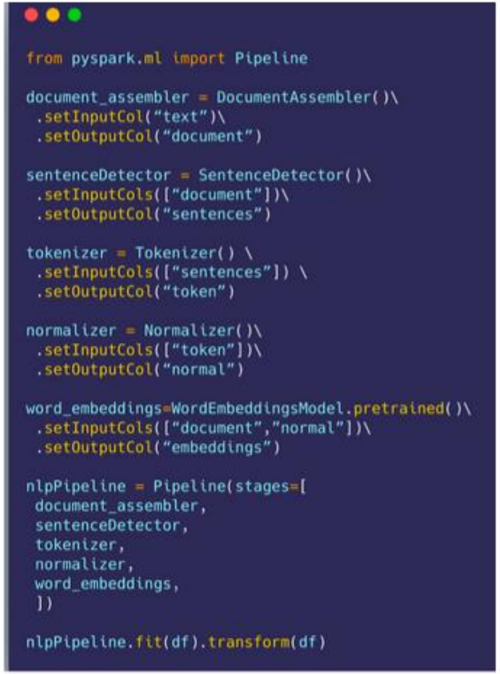
preprocess

vect

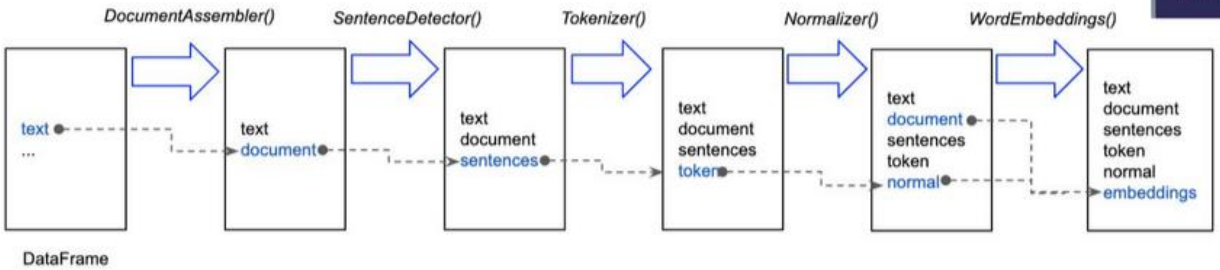
tfidf

model

Fitted Pipeline Model



```
from pyspark.ml import Pipeline

document_assembler = DocumentAssembler()\
 .setInputCol(“text”)\
 .setOutputCol(“document”)

sentenceDetector = SentenceDetector()\
 .setInputCols([“document”])\
 .setOutputCol(“sentences”)

tokenizer = Tokenizer() \
 .setInputCols([“sentences”]) \
 .setOutputCol(“token”)

normalizer = Normalizer()\
 .setInputCols([“token”])\
 .setOutputCol(“normal”)

word_embeddings=WordEmbeddingsModel.pretrained()\
 .setInputCols([“document”,“normal”])\
 .setOutputCol(“embeddings”)

nlpPipeline = Pipeline(stages=[
 document_assembler,
 sentenceDetector,
 tokenizer,
 normalizer,
 word_embeddings,
 ])

nlpPipeline.fit(df).transform(df)
```



DocumentAssembler()

SentenceDetector()

Tokenizer()

Normalizer()

WordEmbeddings()

text
...

text
document

text
document
sentences

text
document
sentences
token

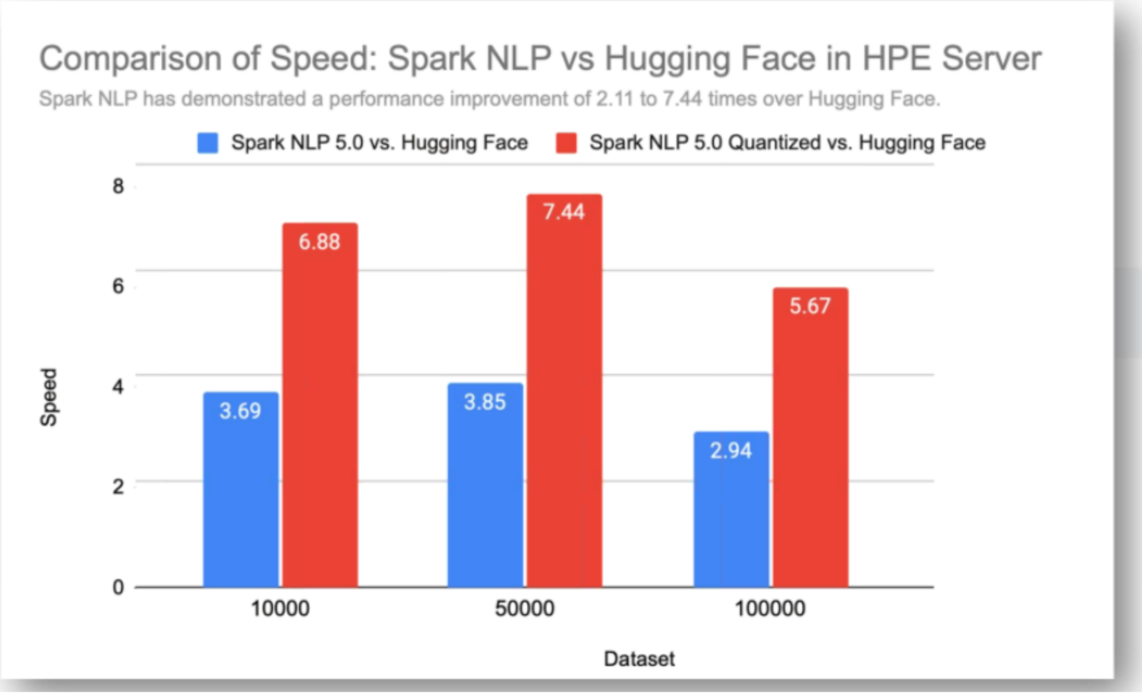
text
document
sentences
token
normal

text
document
sentences
token
normal
embeddings

DataFrame

Calculate Embeddings 3x-7x Faster Than Hugging Face on a Single Server

- HPE Server:
  - AMD EPYC 7542 32-Core Processor
  - 80G memory
- Spark NLP based on Onyx Runtime vs. Hugging Face based on PyTorch

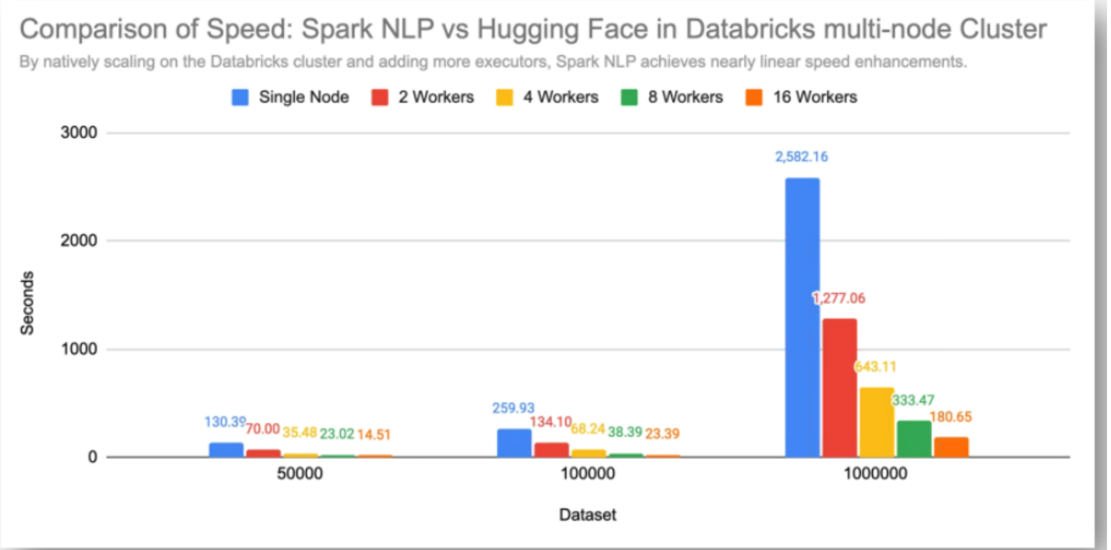


Comparison of Speed: Spark NLP vs Hugging Face in HPE Server

Spark NLP has demonstrated a performance improvement of 2.11 to 7.44 times over Hugging Face.

Scale Up With Zero Code Changes

- Databricks Single Node Cluster
  - 13.0 ML (includes Apache Spark 3.4.0, Scala 2.12)
  - c6i.8xlarge
  - 32 Cores
  - 64 GB Memory
- By natively scaling on the Databricks cluster and adding more executors, Spark NLP 5 achieves near-linear speedup.



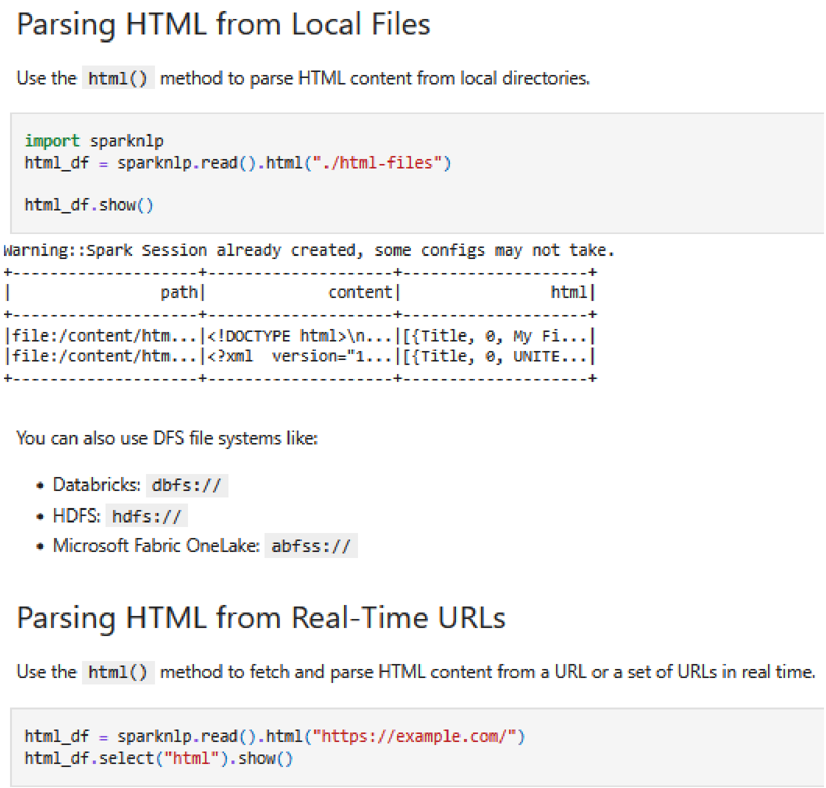
Comparison of Speed: Spark NLP vs Hugging Face in Databricks multi-node Cluster

By natively scaling on the Databricks cluster and adding more executors, Spark NLP achieves nearly linear speed enhancements.

Processing of 1,000,000 records was reduced
from 43 hours to 3 minutes with zero code changes

Transforming Unstructured Data

- Spark NLP 5.2 offers tools and platforms designed to transform unstructured data—such as PDFs, HTML files, emails, Word documents, and images—into formats suitable for use with RAG/LLM and other AI applications – atscale, privately and free.
- The solutions aim to streamline the process of making complex data AI-ready, facilitating easier integration into various machine learning workflows.



Parsing HTML from Local Files

Use the `html()` method to parse HTML content from local directories.

```python
import sparknlp
html_df = sparknlp.read().html("./html-files")

html_df.show()
```

```
Warning::Spark Session already created, some configs may not take.
+--------------------+--------------------+--------------------+
|                path|             content|                html|
+--------------------+--------------------+--------------------+
|file:/content/htm...|<!DOCTYPE html>\n...|[{Title, 0, My Fi...|
|file:/content/htm...|<?xml  version="1...|[{Title, 0, UNITE...|
+--------------------+--------------------+--------------------+
```

You can also use DFS file systems like:

- Databricks: `dbfs://`
- HDFS: `hdfs://`
- Microsoft Fabric OneLake: `abfss://`

Parsing HTML from Real-Time URLs

Use the `html()` method to fetch and parse HTML content from a URL or a set of URLs in real time.

```python
html_df = sparknlp.read().html("https://example.com/")
html_df.select("html").show()
```