Day 1: April 15, 2023 – 9:00 to 13:00 PST

Day 2: April 16, 2023 – 9:00 to 13:00 PST

Day 1: April 17, 2023 – 9:00 to 13:00 PST

Day 2: April 18, 2023 – 9:00 to 13:00 PST

4 days, 16 hrs live training
State-Of-The-Art NLP

Certification exams in 2 weeks

John Snow LABS

Spark NLP for Data Scientists

April 15 & 16, 2024

David Amore Cecchini
cecchini@johnsnowlabs.com

Cabir Celik
cabir@johnsnowlabs.com

Meryem Vildan Sarikaya
meryem@johnsnowlabs.com

Samed Kocer
samed@johnsnowlabs.com

Welcome - We have a lot of things ahead of us

Day-1	50 min	- Introduction to John Snow Labs, Spark NLP and NLP Theory - Spark NLP Pretrained Pipelines
	10 min	Break
	50 min	- Text Preprocessing - Pretrained models in Spark NLP
	10 min	Break
	50 min	- Named Entity Recognition (NER) - ZeroShotNER - Text Classification
		Break
	50 min	- NER Training - Text Classification Training

Welcome - We have a lot of things ahead of us

Day-2	50 min	- Summarization - Question-Answering
	10 min	Break
	50 min	- Automatic Speech Recognition - Image Classification - Neural Machine Translation
	10 min	Break
	50 min	- Splitting documents into meaningful chunks - Ranking retrieved documents - RAG
	10 min	Break
	50 min	- OpenIAI in Spark NLP - Import Transformers into Spark NLP

Session 1 (Day 1)

- NLP Theory and Introduction to John Snow Labs
- Pretrained Pipelines
- Spark, JVM and Spark NLP basic concepts

Spark NLP
for Data Scientists

John Snow LABS

Resources

CODE:

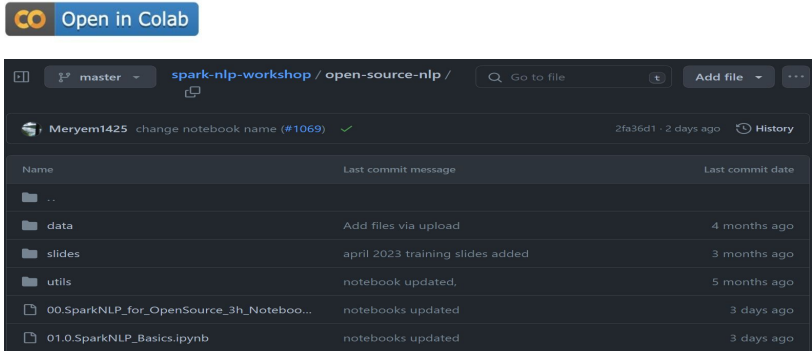
- https://github.com/JohnSnowLabs/spark-nlp-workshop/tree/master/open-source-nlp

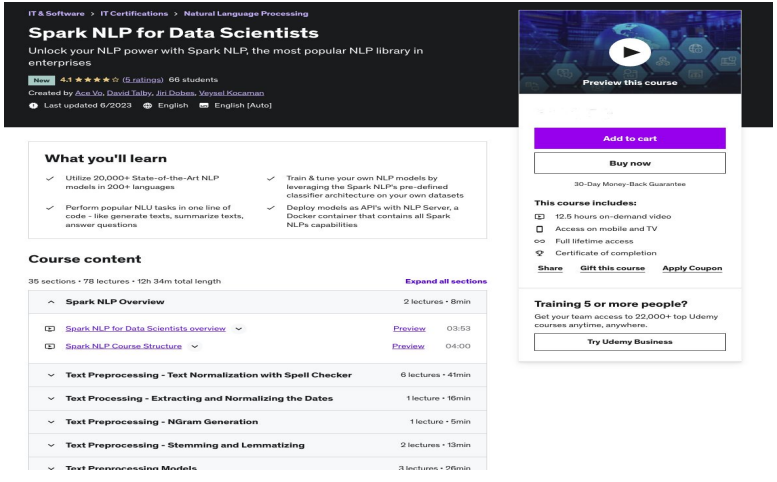
BOOKMARK:

- https://sparknlp.org/
- https://sparknlp.org/docs/en/quickstart
- https://sparknlp.org/api/python/reference/index.html
- Spark-nlp.slack.com

MOOC:

- https://www.udemy.com/course/spark-nlp-for-data-scientists/
- https://github.com/JohnSnowLabs/spark-nlp-workshop/tree/master/Spark_NLP_Udemy_MOOC/Open_Source



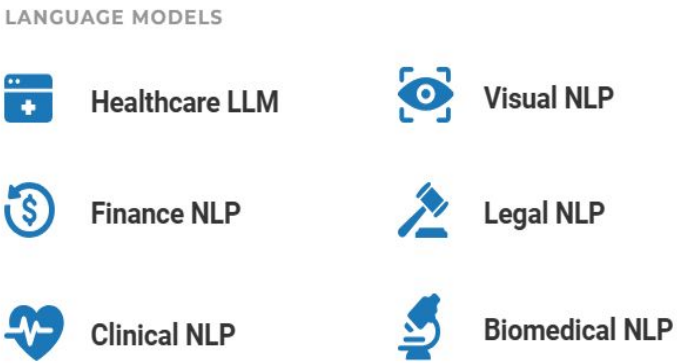


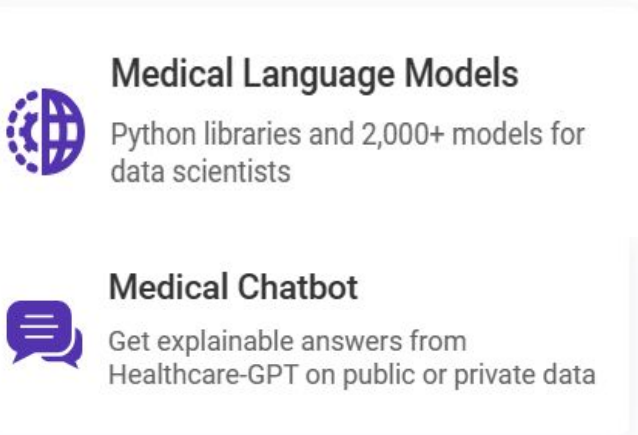
Introduction

John Snow Labs's Products

Licensed products:

State of the art models and libraries for specific domains: Healthcare, Finance, Legal, and Computer Vision.





John Snow Labs's Products

Free or open-source tools:

General NLP and no-code software for data scientists and practitioners

Generative AI Lab

Train, tune, and share custom AI models without coding

Spark NLP

State-of-the-art natural language processing at scale

Responsible AI with LangTest

Deliver safe and effective language models, by generating and running 100+ test types

John Snow Labs's Products

In this training:

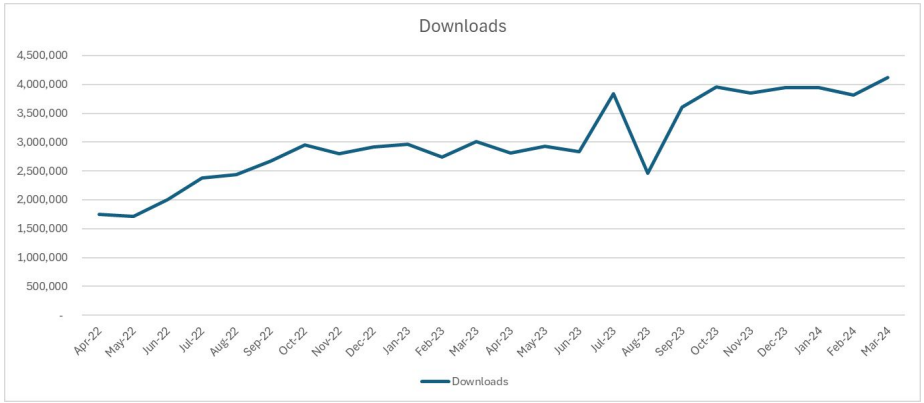
- Spark NLP for Data Scientists: April 15 & 16
  - Open-source NLP library on top of Spark and Spark MLlib
  - Base annotators for building pipelines

- Healthcare NLP for Data Scientists: April 17 & 18
  - Healthcare-specific models and applications
  - Healthcare LLM

Spark NLP

Introducing Spark NLP

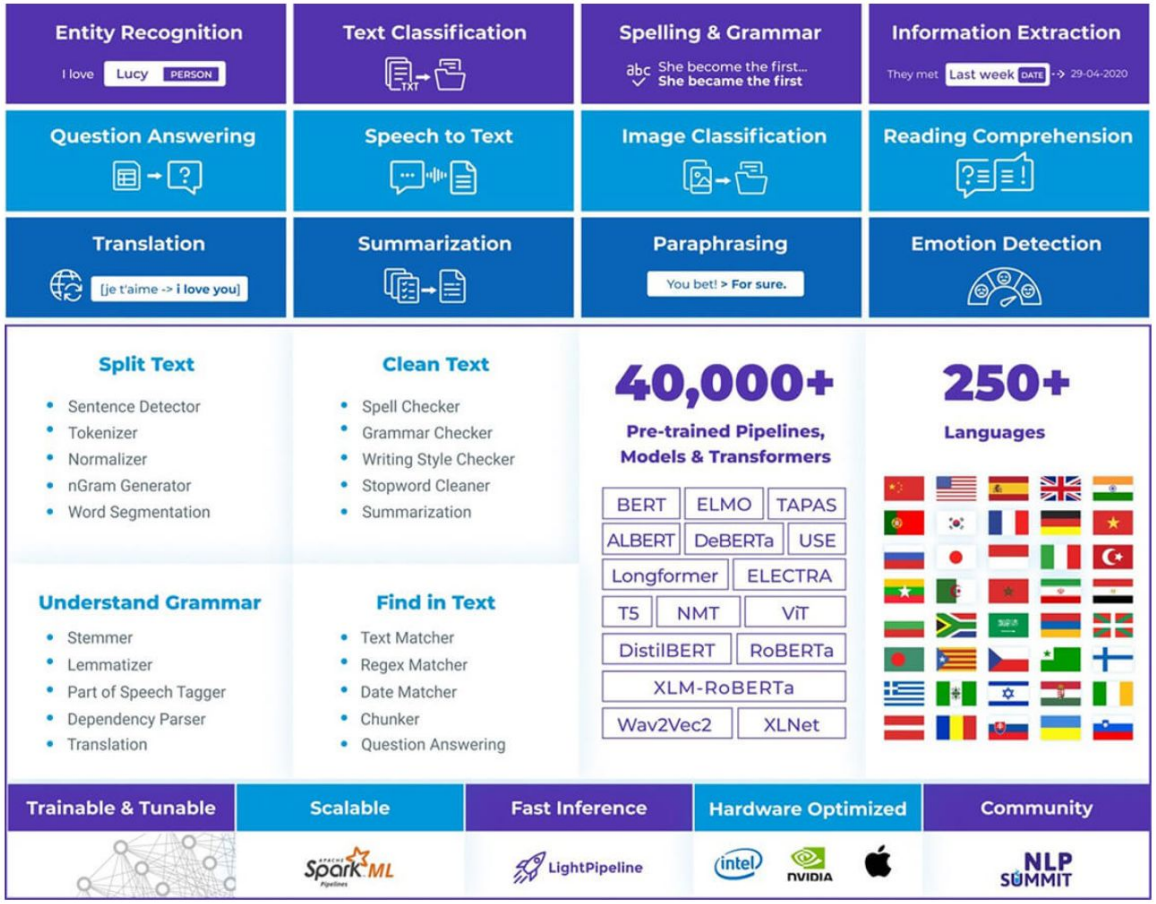
https://www.pepy.tech/projects/spark-nlp



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

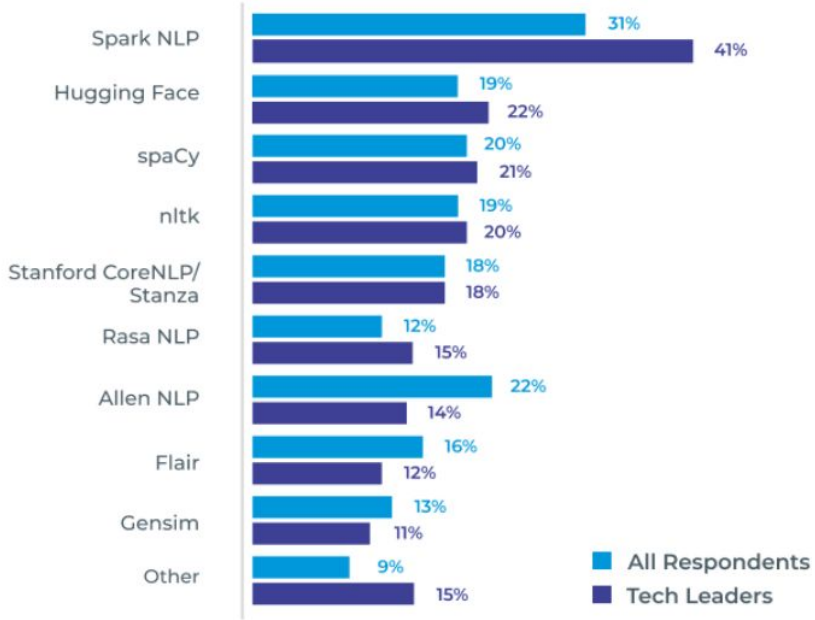
- A single unified solution for all your NLP needs
- Take advantage of transfer learning and implementing the latest and greatest **SOTA** algorithms and models in NLP research
- The most widely used NLP library in industry (5 years in a row)
- The most scalable, accurate and fastest library in NLP history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

Spark NLP Modules

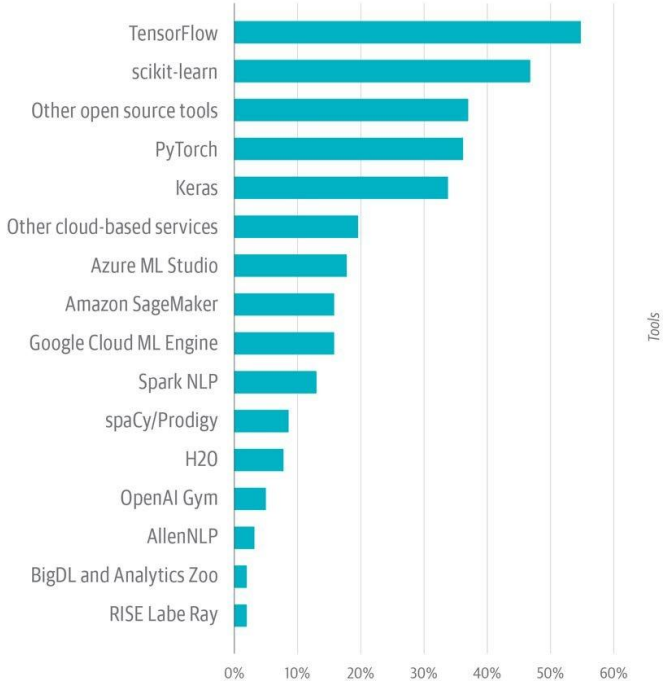


Spark NLP in Industry

Which NLP libraries does your organization use?



Which of the following AI tools do you use?



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

TRUSTED BY


Microsoft


Google


amazon


intel


IBM


databricks


verizon


indeed


CapitalOne


viacom


McKESSON


MERCK


Roche


selectdata


UiPath
Reboot Work.


ASCO
CANCER·LINQ
DISCOVERY


cnrs


Imperial College
London


Georgia
Tech


STANFORD
UNIVERSITY

Biomedical Named Entity Recognition at Scale

Veysel Kocaman
John Snow Labs Inc.
16192 Coastal Highway
Lewes, DE , USA 19958
veysel@johnsnowlabs.com

***Abstract*—Named entity recognition (NER) is a widely applicable natural language processing task and building block of question answering, topic modeling, information retrieval, etc. In the medical domain, NER plays a crucial role by extracting meaningful chunks from clinical notes and reports, which are then fed to downstream tasks like assertion status detection, entity resolution, relation extraction, and de-identification. Reimplementing a Bi-LSTM-CNN-Char deep learning architecture on top of Apache Spark, we present a single trainable NER model that obtains new state-of-the-art results on seven public biomedical benchmarks without using heavy contextual embeddings like BERT. This includes improving BC4CHEMD to 93.72% (4.1% gain), Species800 to 80.91% (4.6% gain), and JNLPBA to 81.29% (5.2% gain). In addition, this model is freely available within a production-grade code base as part of the open-source Spark NLP library; can scale up for training and inference in any Spark cluster; has GPU support and libraries for popular programming languages such as Python, R, Scala and Java; and can be extended to support other human languages with no code changes.**

I. Introduction

Electronic health records (EHRs) are the primary source of information for clinicians tracking the care of their patients. Information fed into these systems may be found in structured fields for which values are inputted electronically (e.g. laboratory test orders or results) [1] but most of the time information in these records is unstructured making it largely inaccessible

Spark NLP: Natural Language Understanding at Scale

Veysel Kocaman, David Talby

John Snow Labs Inc.
16192 Coastal Highway
Lewes, DE , USA 19958
eysel, david}@johnsnowlabs.com

Accurate Clinical and Biomedical Named Entity Recognition at Scale

Anonymous NAACL-HLT 2021 submission

Abstract

Named entity recognition (NER) is one of the most important building blocks of NLP tasks in the medical domain by extracting meaningful chunks from clinical notes and reports, which are then fed to downstream tasks like assertion status detection, entity resolution, relation extraction, and de-identification. Due to the growing volume of healthcare data in unstructured format, an increasingly important challenge is providing high accuracy implementations of state-of-the-art deep learning (DL) algorithms at scale. In this study, we introduce a production-grade clinical and biomedical NER algorithm based on a modified BiLSTM-CNN-Char DL architecture built on top of Apache Spark. This algorithm establishes new state-of-the-art accuracy on 7 of 8 well-known biomedical NER benchmarks and 3 clinical concept extraction challenges: 2010 i2b2/VA clinical concept extraction, 2014 n2c2 de-identification, and 2018 n2c2 medication extraction. Moreover, clinical NER models trained using this implemen-

Improving Clinical Document Understanding on COVID-19 Research with Spark NLP

Veysel Kocaman, David Talby
John Snow Labs Inc.
16192 Coastal Highway
Lewes, DE , USA 19958
{veysel, david}@johnsnowlabs.com

Abstract

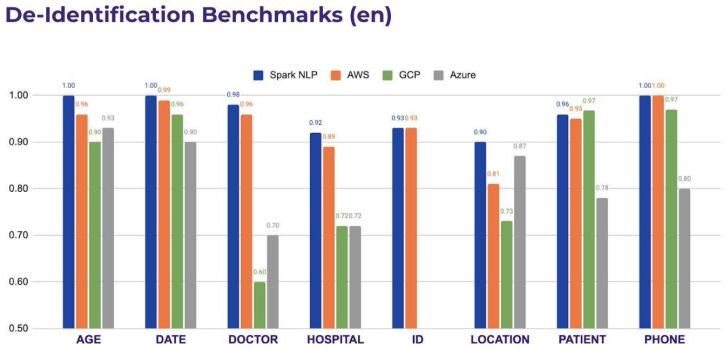
Following the global COVID-19 pandemic, the number of scientific papers studying the virus has grown massively, leading to increased interest in automated literate review. We present a clinical text mining system that improves on previous efforts in three ways. First, it can recognize over 100 different entity types including social determinants of health, anatomy, risk factors, and adverse events in addition to other commonly used clinical and biomedical entities. Second, the text processing pipeline includes assertion status detection, to distinguish between clinical facts that are present, absent, conditional, or about someone other than the patient. Third, the deep learning models used are more accurate than previously available, leveraging an integrated pipeline of state-of-the-art pre-trained named entity recognition models, and improving on the previous best performing benchmarks for assertion status detection. We illustrate extracting trends and insights - e.g. most frequent disorders and symptoms, and most common vital signs and EKG findings – from the COVID-19 Open Research Dataset (CORD-19). The system is built using the Spark NLP library which natively supports scaling to use distributed clusters, leveraging GPU's, configurable and reusable NLP pipelines, healthcare-specific embeddings, and the ability to train models to support new entity types or human languages with no code changes.

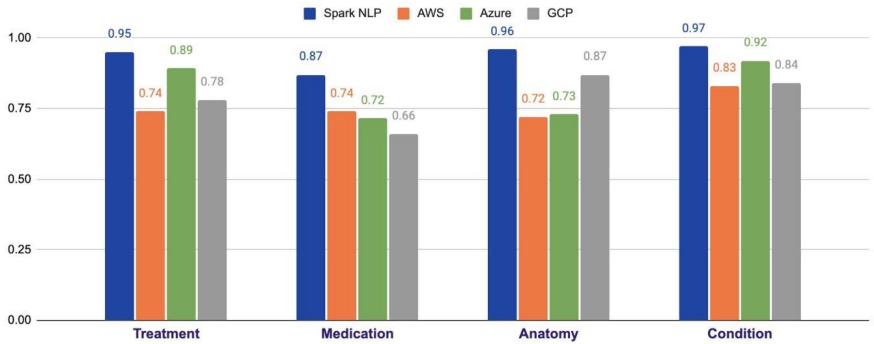
be found in structured fields for which values are inputted electronically (e.g. laboratory test orders or results) (Liede et al. 2015) but most of the time information in these records is unstructured making it largely inaccessible for statistical analysis (Murdoch and Detsky 2013). These records include information such as the reason for administering drugs, previous disorders of the patient or the outcome of past treatments, and they are the largest source of empirical data in biomedical research, allowing for major scientific findings in highly relevant disorders such as cancer and Alzheimer's disease (Perera et al. 2014).

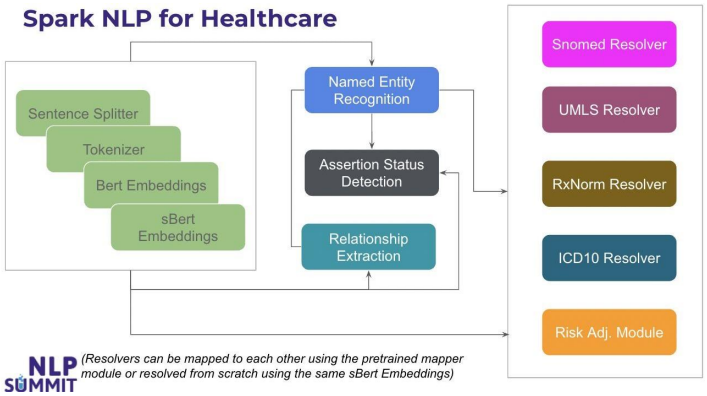
A primary building block in such text mining systems is named entity recognition (NER) - which is regarded as a critical precursor for question answering, topic modelling, information retrieval, etc (Yadav and Bethard 2019). In the medical domain, NER recognizes the first meaningful chunks out of a clinical note, which are then fed down the processing pipeline as an input to subsequent downstream tasks such as clinical assertion status detection (Uzuner et al. 2011), clinical entity resolution (Tzitzivacos 2007) and de-identification of sensitive data (Uzuner, Luo, and Szolovits 2007) (see Figure 1). However, segmentation of clinical and drug entities is considered to be a difficult task in biomedical NER systems because of complex orthographic structures of named entities

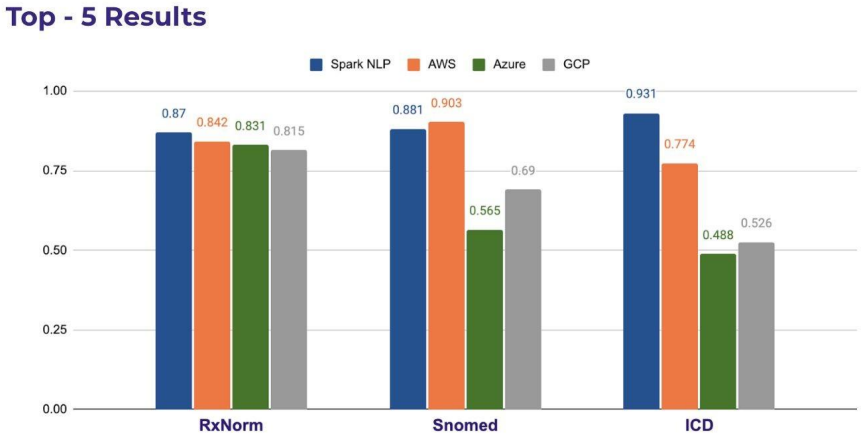
Peer-reviewed papers on
Spark NLP NER

Most Accurate in the Industry









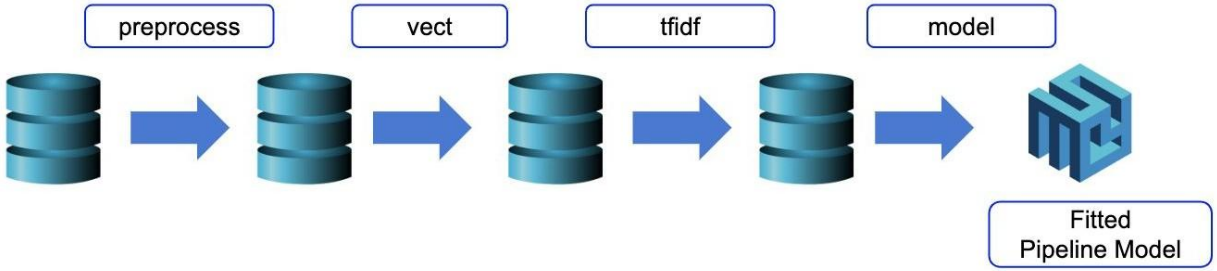
[Spark NLP vs Commercial healthcare NLP Solutions Benchmark]

General Concepts

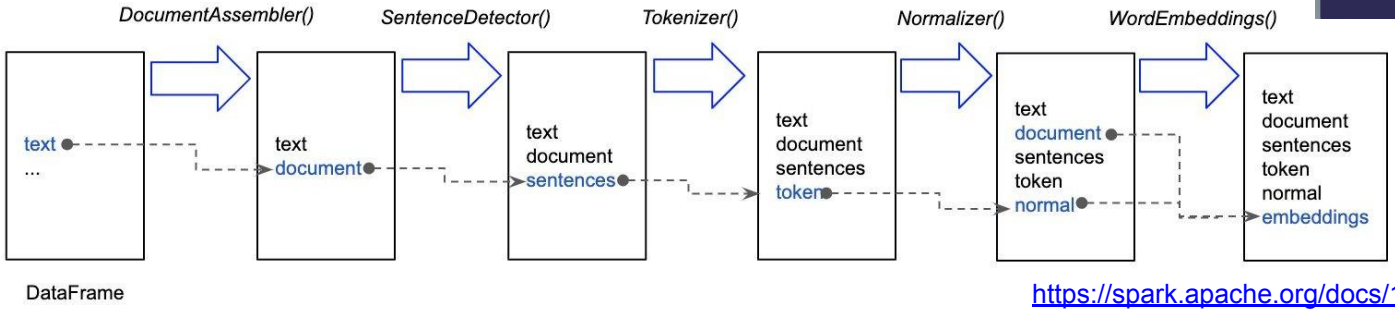
01	**Annotator Approach**	• Similar to Estimators in Spark MLlib • Needs to be fit to data (becoming and annotator model) • Trains NLP models
02	**Annotator Model**	• Similar to Transformers in Spark MLlib • Applies transformations to Spark Data Frames
03	**Pipelines**	• Same as pipelines in Spark MLlib • Chains different annotators to keep all stages in one object • *Pipeline* applies to Spark Data Frames, *LightPipeline* applies to strings or list of strings
02	**Annotation**	• Structured data with the following fields ○ *annotatorType, begin, end, result, metadata, and Embeddings* (new in v2.0)

Spark Pipelines

Pipeline of annotators




```
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)
```



https://spark.apache.org/docs/1.6.0/ml-guide.html#pipeline-components

Light Pipelines for Fast Inference

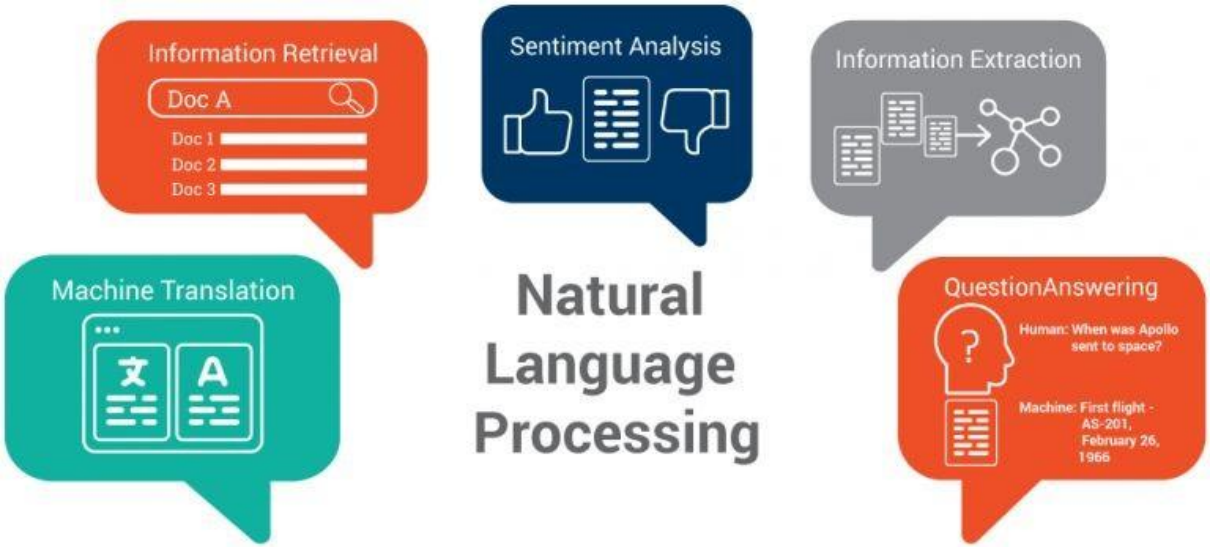
Spark is like a locomotive racing a bicycle. The bike will win if the load is light, it is quicker to accelerate and more agile, but with a heavy load the locomotive might take a while to get up to speed, but it's going to be faster in the end.

Faster inference

```
from sparknlp.base import LightPipeline
LightPipeline(someTrainedPipeline).annotate(someStringOrArray)
```

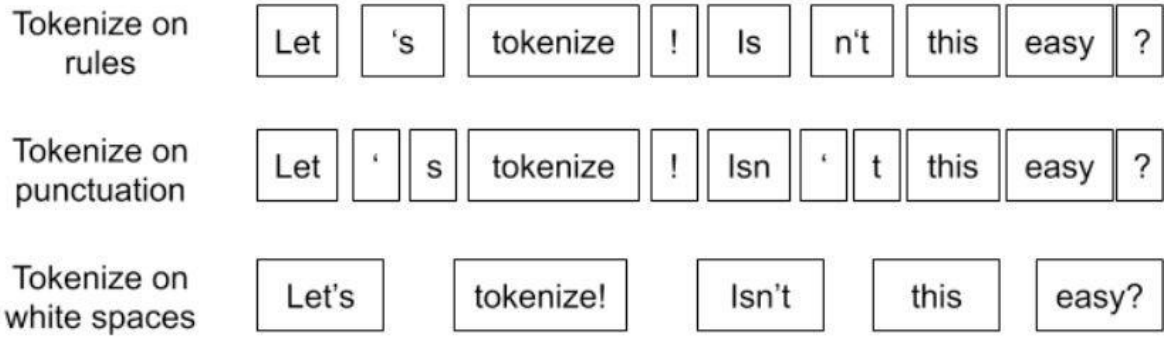
LightPipelines are Pipelines converted into a single machine but multithreaded task, becoming more than 10x times faster for smaller amounts of data (small is relative, but 50k sentences is roughly a good maximum).

NLP Basics


Information Retrieval
Doc A
Doc 1
Doc 2
Doc 3
Sentiment Analysis
Information Extraction
Machine Translation
Natural Language Processing
QuestionAnswering
Human: When was Apollo sent to space?
Machine: First flight - AS-201, February 26, 1966

NLP Basics

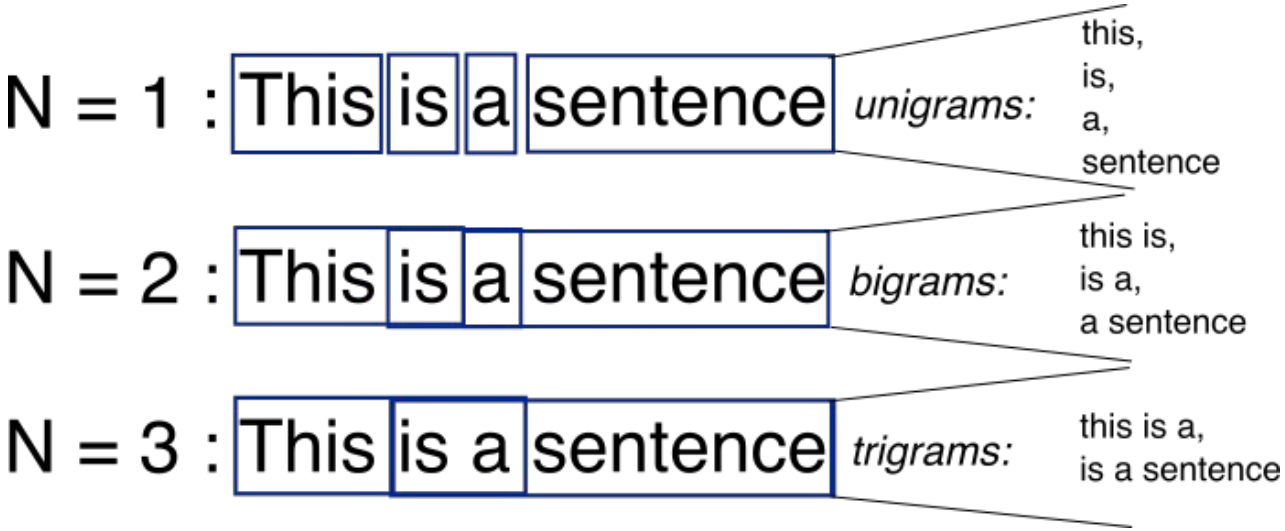
Tokenization



- First step in NLP pipelines, raw text is broken down into smaller, more manageable pieces. Without tokenization, NLP models would have to process the text as one large unit.
- Many other tokenization methods (character, sub-word, sentence).

NLP Basics

N-gram Tokenization

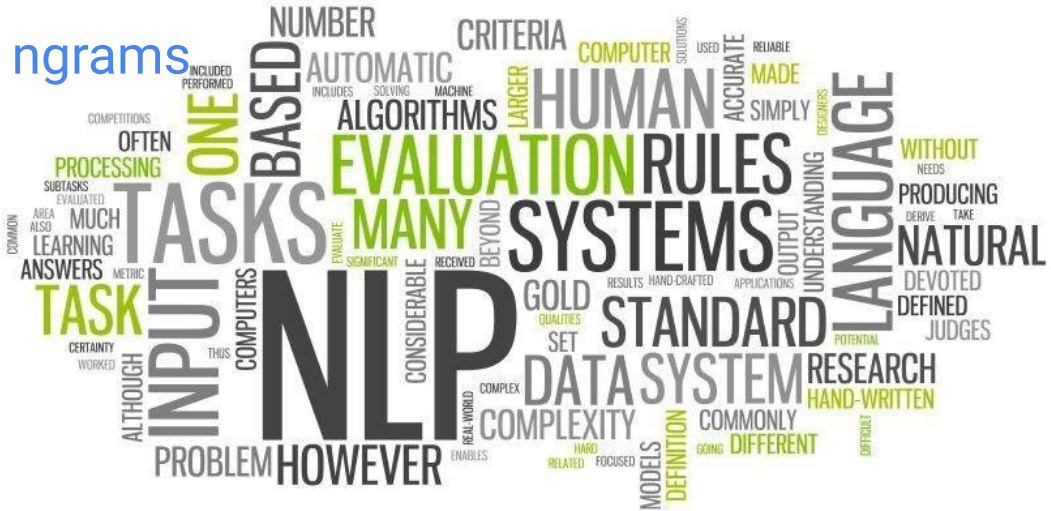


- Kind of tokenizers which split words or sentences into several tokens
- Each token has certain number of words
- Number of words depends on the type of n-gram tokenizer
- Unigram, bigram, trigram, etc.

NLP Basics

Unsupervised Keyword Extraction

YAKE! Is Yet Another Keyword Extraction Algorithm that can extract keywords without any weight by leveraging statistical properties of ngrams



Notebook 08.0
YAKE

Pre-Processing

NLP Basics

LEMMATIZATION

Find the **lemma** of each word:

- How does it show in the dictionary?

Uses a lookup from a full dictionary.

```
am, are, is    →    be

liver          →    liver

lives          →    live
```

STEMMING

Find the **stem** of each word.

Uses rules: e.g, remove common suffixes.

Form	Suffix	Stem
studies	-es	studi
studying	-ing	study
niñas	-as	niñ
niñez	-ez	niñ

- The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form for normalization purposes.
- Lemmatization always returns real words, stemming doesn't.

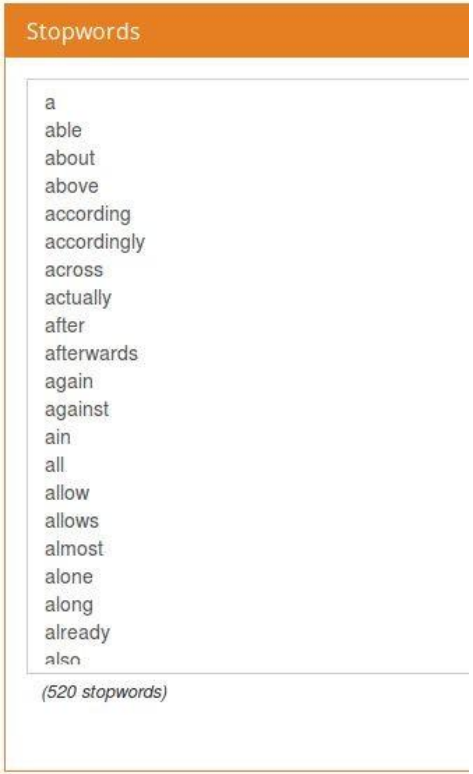
Pre-Processing

NLP Basics

Stopword Removal

it was a bright cold day in april and the clocks were striking thirteen winston smith his chin nuzzled into his breast in an effort to escape the vile wind slipped quickly through the glass doors of victory mansions though not quickly enough to prevent a swirl of gritty dust from entering along with him

bright cold day april clocks striking thirteen winston smith chin nuzzled breast effort escape vile wind slipped quickly glass doors victory mansions though quickly enough prevent swirl gritty dust entering along



- For tasks like text classification, where the text is to be classified into different categories, **stopwords** are **removed** or excluded from the given text so that more focus can be given to those words which define the meaning of the text. Stopwords are considered insignificant in this case.

Pre-Processing

NLP Basics

Spell Checking and Correction


```
val pipeline = PretrainedPipeline("spell_check_ml", "en")
val result = pipeline.annotate("Harry Potter is a graet muvie")

println(result("spell"))
/* will print Seq[String](…, "is", "a", "great", "movie") */
```

- 3 trainable approaches
- **Norvig Approach**:
  - Retrieves tokens and auto-corrects based on a given dictionary
- **Symmetric Delete**:
  - Uses distance metrics to find possible words
- **Context Aware**:
  - Most accurate: Judges words in context
  - Deep learning based

Pre-Processing

NLP Basics

Normalization

Remove or replace undesirable characters or regular expressions:

from: `@Have a$ #2great birth) day>!`

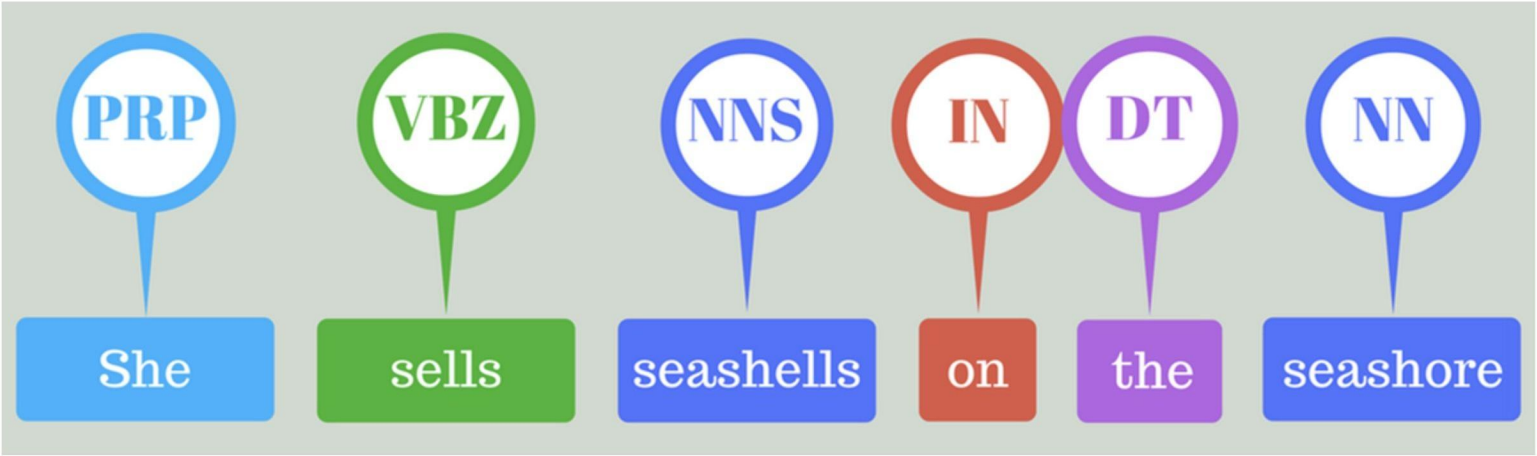
to: `Have a great birth day!`

Spark NLP also comes with a Slang normalizer:

NLP Basics

Part Of Speech Tagging

Often useful for recognizing named entities or word relationships.

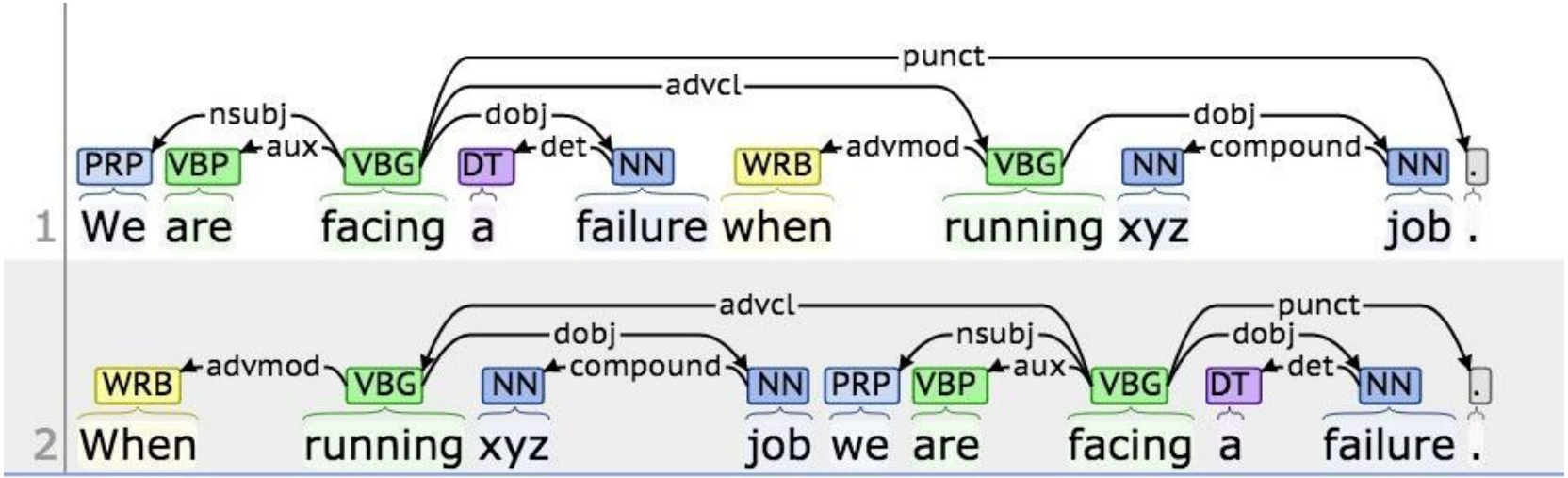


A **POS tag** (or **part-of-speech tag**) is a special label assigned to each token (word) in a text corpus to indicate the **part of speech** and often also other grammatical categories such as tense, number (plural/singular), case etc.

NLP Basics

Dependency Parsing

Useful for extracting relationships (i.e. building knowledge graphs):



Feature-Engineering & Classification

NLP Basics

Named Entity Recognition (NER)

NER is a subtask of information extraction that seeks to locate and classify named entity mentioned in unstructured text into pre-defined categories such as person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages, etc.

But Google ORG is starting from behind. The company made a late push into hardware, and Apple ORG 's Siri PRODUCT , available on iPhones PRODUCT , and Amazon ORG 's Alexa PRODUCT software, which runs on its Echo PRODUCT and Dot PRODUCT devices, have clear leads in consumer adoption.

Feature-Engineering & Classification

NLP Basics

Rule-based Entity Recognition with *EntityRuler*

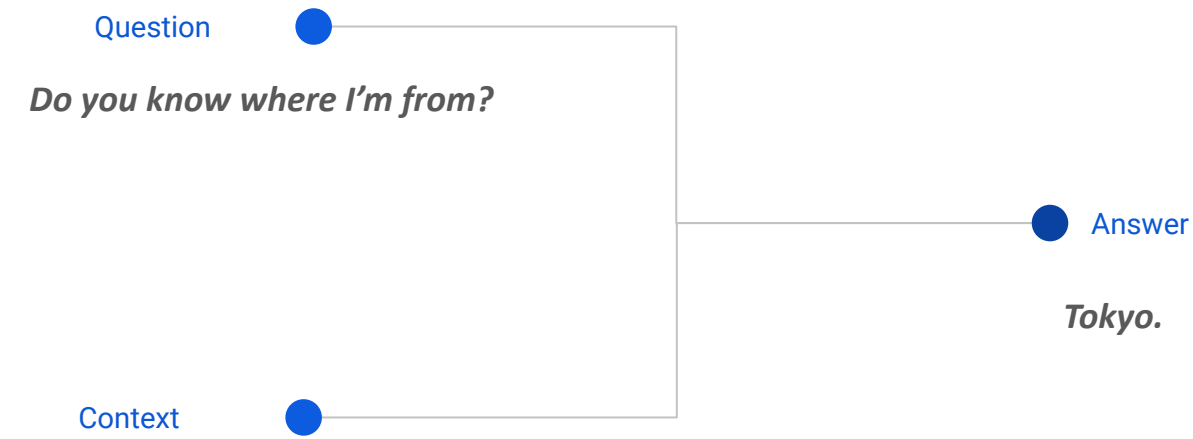
Fits an Annotator to match exact strings or regex patterns provided in a file against a Document and assigns them an named entity.

```
[
    {
      "id": "person-regex",
      "label": "PERSON",
      "patterns": ["\w+\s\w+", "\w+-\w+"]
    },
    {
      "id": "locations-words",
      "label": "LOCATION",
      "patterns": ["Winterfell"]
    }
]
```

```
{"id": "names-with-j", "label": "PERSON", "patterns": ["Jon", "John", "John Snow"]}
{"id": "names-with-s", "label": "PERSON", "patterns": ["Stark", "Snow"]}
{"id": "names-with-e", "label": "PERSON", "patterns": ["Eddard", "Eddard Stark"]}
```

NLP Basics

Question Answering



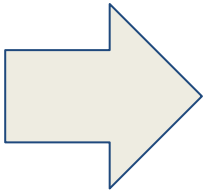
LLM

NLP Basics

Summarization

(Reuters) - Mastercard Inc said on Wednesday it was planning to offer support for some cryptocurrencies on its network this year, joining a string of big-ticket firms that have pledged similar support.
The credit-card giant's announcement comes days after Elon Musk's Tesla Inc revealed it had purchased $1.5 billion of bitcoin and would soon accept it as a form of payment.
Asset manager BlackRock Inc and payments companies Square and PayPal have also recently backed cryptocurrencies.
Mastercard already offers customers cards that allow people to transact using their cryptocurrencies, although without going through its network.

...



The world's largest credit - Mastercard giant Mastercard has announced it will begin to support for some of the digital currency.

LLM

NLP Basics

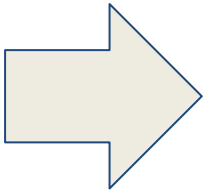
Machine Translation

LLM

NLP Basics

Text Generation

Tell me a dad joke.



Ok, here is a dad joke:

What do you call a factory that makes okay products?

A satisfactory.

LLM

Session 1 (Day 1) - Coding Time

- Notebook 1 - Spark NLP Basics

Spark NLP
for Data Scientists

John Snow LABS

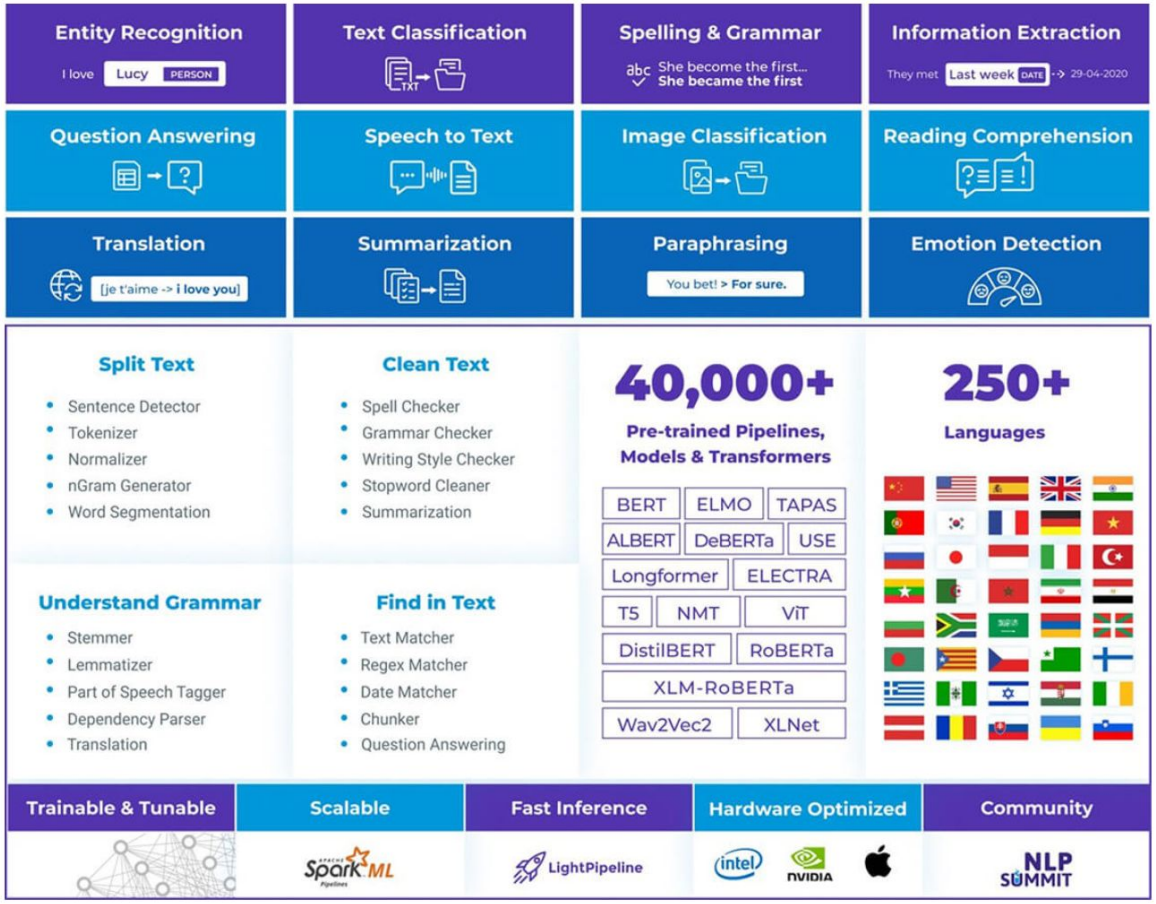
Session 2 (Day 1)

- Text Preprocessing
- Pretrained models

Spark NLP
for Data Scientists

John Snow LABS

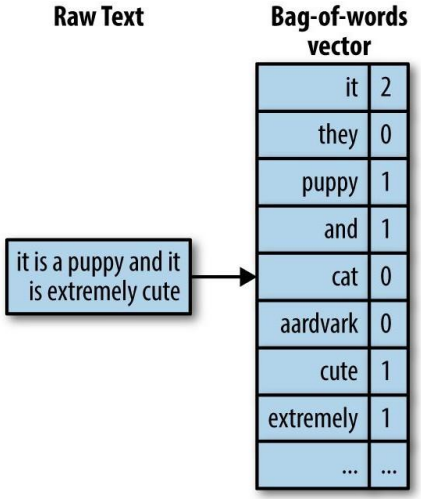
Spark NLP Modules

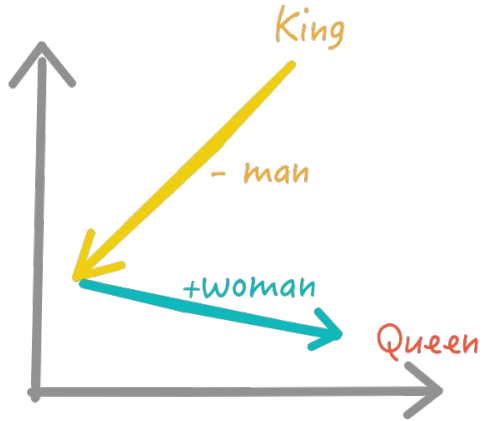


Word & Sentence Embeddings

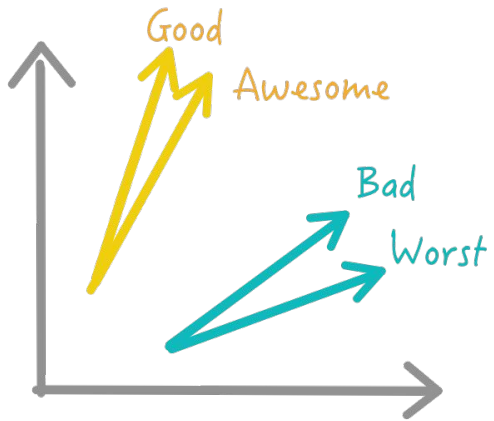
```
In [9]: doc[3].vector
Out[9]: array([ 0.037103 , -0.31259  , -0.17857  ,  0.30001  ,  0.078154 ,
                0.17958  ,  0.12048  , -0.11879  , -0.20601  ,  1.2849   ,
               -0.20409  ,  0.80613  ,  0.34344  , -0.19191  , -0.084511 ,
                0.17339  ,  0.042483 ,  2.0282   , -0.16278  , -0.60306  ,
               -0.53766  ,  0.35711  ,  0.22882  ,  0.1171   ,  0.42983  ,
                0.16165  ,  0.407    ,  0.036476 ,  0.52636  , -0.13524  ,
               -0.016897 ,  0.029259 , -0.079115 , -0.32305  ,  0.052255 ,
               -0.3617   , -0.18355  , -0.34717  , -0.3691   ,  0.16881  ,
                0.21018  , -0.38376  , -0.096909 , -0.36296  , -0.37319  ,
                0.0021152,  0.32512  ,  0.063977 ,  0.36249  , -0.26935  ,
               -0.59341  , -0.13625  ,  0.016425 , -0.2474   , -0.07498  ,
                0.034708 , -0.01476  , -0.11648  ,  0.25559  , -0.35002  ,
               -0.52707  ,  0.21221  ,  0.062456 ,  0.26184  ,  0.53149  ,
                0.34957  , -0.22692  ,  0.44076  ,  0.4438   ,  0.6335   ,
               -0.049757 , -0.08134  ,  0.65618  , -0.4716   ,  0.090675 ,
               -0.084873 ,  0.31455  , -0.38495  , -0.19247  ,  0.48064  ,
                0.26688  ,  0.095743 ,  0.13024  ,  0.37023  ,  0.46269  ,
               -0.32844  ,  0.17375  , -0.36325  ,  0.30672  , -0.075042 ,
               -0.64684  , -0.49822  ,  0.12372  , -0.28547  ,  0.61811  ,
               -0.19228  ,  0.0040473,  0.1774   ,  0.033154 , -0.54862  ,
                0.34695  , -0.53506  , -0.013381 ,  0.085712 , -0.054447 ,
               -0.64673  ,  0.016749 ,  0.47676  ,  0.037803 , -0.10066  ,
               -0.4165   , -0.20252  ,  0.2794   ,  0.10852  , -0.40154  ,
```

- Neural Networks and most other algorithms don't work well with strings
  - Why not convert a string into a vector with numbers!
  - Semantically similar words vectors should be **close** in the vector space learned by the
  - Dissimilar words are **far** from each other in vector space
  - Usually high dimensional, to give model more space to encode semantics
- Useful for many downstream tasks and commonly used in NLP
- Reduce dimensionality of the input space





a) Learns Analogy



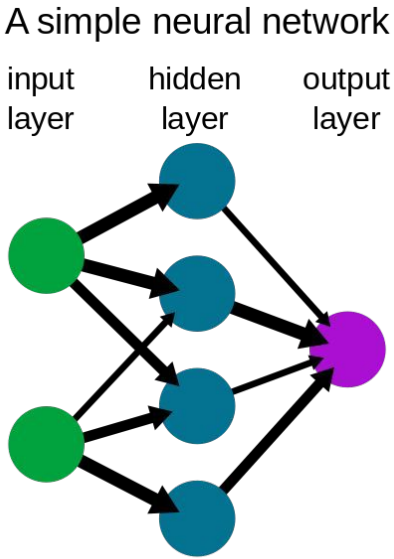
b) Similar Words have same angles

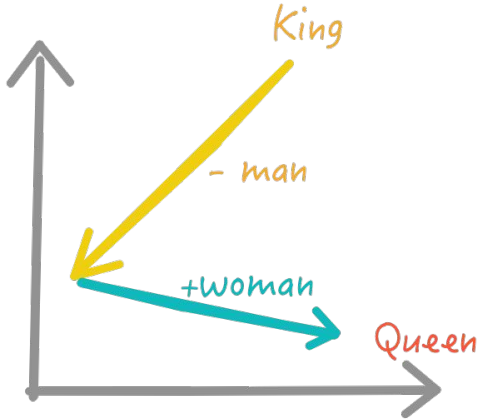
Feature-Engineering

Word & Sentence Embeddings

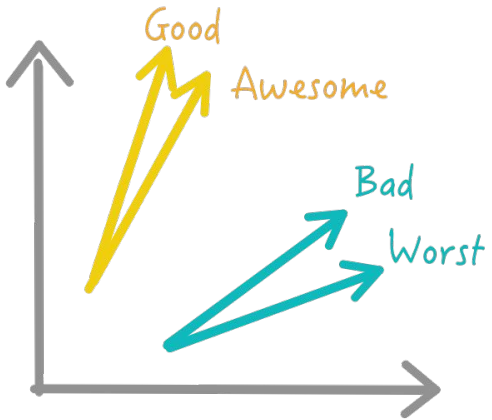
```
In [9]: doc[3].vector
Out[9]: array([ 0.037103 , -0.31259  , -0.17857  ,  0.30001  ,  0.078154 ,
                0.17958  ,  0.12048  , -0.11879  , -0.20601  ,  1.2849   ,
               -0.20409  ,  0.80613  ,  0.34344  , -0.19191  , -0.084511 ,
                0.17339  ,  0.042483 ,  2.0282   , -0.16278  , -0.60306  ,
               -0.53766  ,  0.35711  ,  0.22882  ,  0.1171   ,  0.42983  ,
                0.16165  ,  0.407    ,  0.036476 ,  0.52636  , -0.13524  ,
               -0.016897 ,  0.029259 , -0.079115 , -0.32305  ,  0.052255 ,
               -0.3617   , -0.18355  , -0.34717  , -0.3691   ,  0.16881  ,
                0.21018  , -0.38376  , -0.096909 , -0.36296  , -0.37319  ,
                0.0021152,  0.32512  ,  0.063977 ,  0.36249  , -0.26935  ,
               -0.59341  , -0.13625  ,  0.016425 , -0.2474   , -0.07498  ,
                0.034708 , -0.01476  , -0.11648  ,  0.25559  , -0.35002  ,
               -0.52707  ,  0.21221  ,  0.062456 ,  0.26184  ,  0.53149  ,
                0.34957  , -0.22692  ,  0.44076  ,  0.4438   ,  0.6335   ,
               -0.049757 , -0.08134  ,  0.65618  , -0.4716   ,  0.090675 ,
               -0.084873 ,  0.31455  , -0.38495  , -0.19247  ,  0.48064  ,
                0.26688  ,  0.095743 ,  0.13024  ,  0.37023  ,  0.46269  ,
               -0.32844  ,  0.17375  , -0.36325  ,  0.30672  , -0.075042 ,
               -0.64684  , -0.49822  ,  0.12372  , -0.28547  ,  0.61811  ,
               -0.19228  ,  0.0040473,  0.1774   ,  0.033154 , -0.54862  ,
                0.34695  , -0.53506  , -0.013381 ,  0.085712 , -0.054447 ,
               -0.64673  ,  0.016749 ,  0.47676  ,  0.037803 , -0.10066  ,
               -0.4165   , -0.20252  ,  0.2794   ,  0.10852  , -0.40154  ,
```

- Deep-Learning-based natural language processing systems encode **words** and **sentences** in fixed-length dense vectors to drastically improve the processing of textual data.
- Based on **The Distributional Hypothesis**: Words that occur in the same contexts tend to have similar meanings.
- Elmo and Bert-family embeddings are context-aware.





a) Learns Analogy

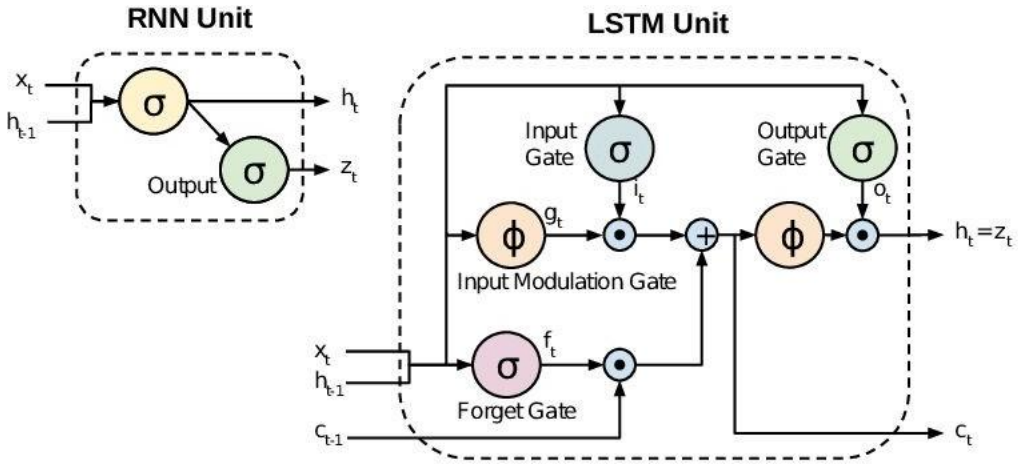


b) Similar Words have same angles

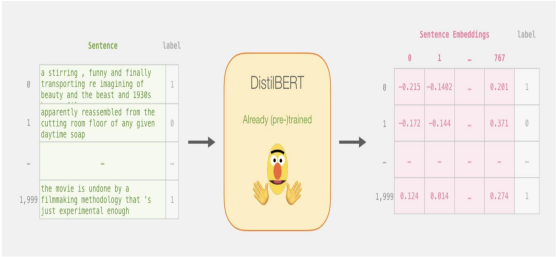
Feature-Engineering

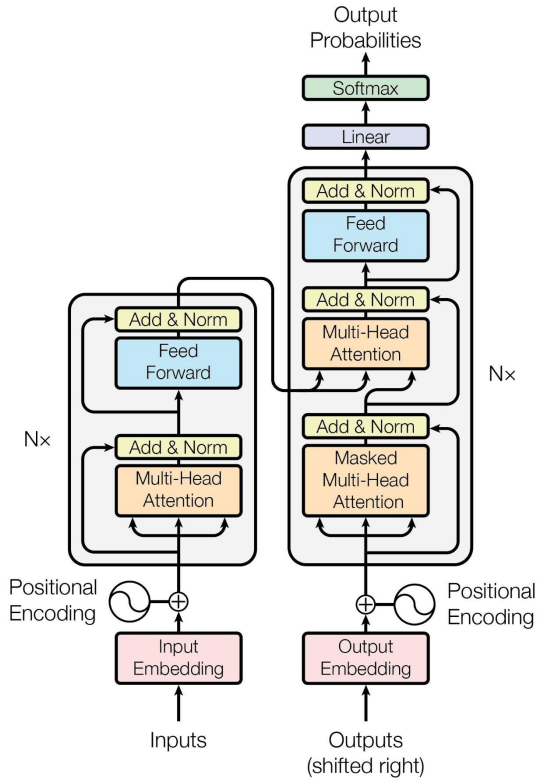
LSTM's and RNN's for Embeddings

- RNN :
  - Slow to train
  - Not well parallelizable
  - Vanishing/Exploding Gradients
- LSTM
  - Fixes vanishing/exploding gradient
  - not scalable
  - No transfer learning



Transformers - Attention is All you Need - **Visualized**
A highlighly parallelizable NN Architecture for Sequential Data





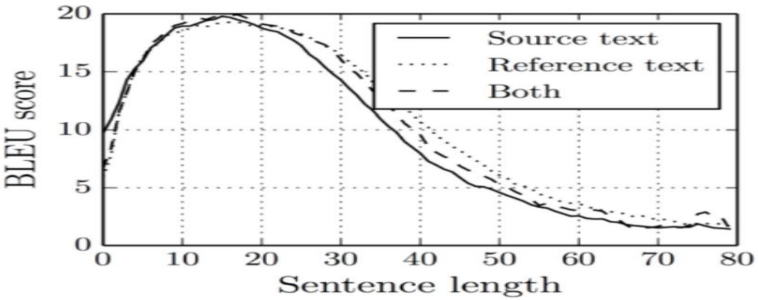
Attention is All You Need

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

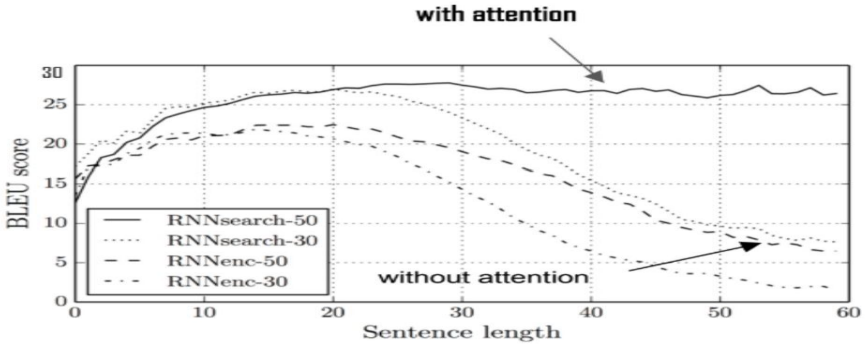
Feature-Engineering

Transformers - Attention is All you Need - Why they rock
A highlighly parallelizable NN Architecture for Sequential Data

NMT with LSTMs



NMT with LSTMs + attention



Bahdanau et al. 2014
Slide: Text generation with attention, GTC 2017, Valentin Malykh (2017)

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Feature-Engineering

100+ Languages supported by [Language-agnostic BERT Sentence Embedding (LABSE)] and [XLM-RoBERTa]

Train in **1 Language**, predict in **100+ different languages**

Notebook 5.2 Training Multilingual Classifier

```
# Binary Class Classifier, 2 classes
nlu.load('xx.embed_sentence.labse train.sentiment').fit(train_df).predict(test_df)

# Multi Class Classifier, N classes
nlu.load('xx.embed_sentence.labse train.classifier').fit(train_df).predict(test_df)

# Multi Class Classifier with multiple labels example (i.e. Hashtags)
# N classes, where one row can be assigned up to N labels
nlu.load('xx.embed_sentence.labse train.multi_classifier').fit(train_df).predict(test_df)
```

ISO	NAME	ISO	NAME	ISO	NAME
af	AFRIKAANS	ht	HAITIAN_CREOLE	pt	PORTUGUESE
am	AMHARIC	hu	HUNGARIAN	ro	ROMANIAN
ar	ARABIC	hy	ARMENIAN	ru	RUSSIAN
as	ASSAMESE	id	INDONESIAN	rw	KINYARWANDA
az	AZERBAIJANI	ig	IGBO	si	SINHALESE
be	BELARUSIAN	is	ICELANDIC	sk	SLOVAK
bg	BULGARIAN	it	ITALIAN	sl	SLOVENIAN
bn	BENGALI	ja	Japanese	sm	SAMOAN
bo	TIBETAN	jv	JAVANESE	sn	SHONA
bs	BOSNIAN	ka	GEORGIAN	so	SOMALI
ca	CATALAN	kk	KAZAKH	sq	ALBANIAN
ceb	CEBUANO	km	KHMER	sr	SERBIAN
co	CORSICAN	kn	KANNADA	st	SESOTHO
cs	CZECH	ko	KOREAN	su	SUNDANESE
cy	WELSH	ku	KURDISH	sv	SWEDISH
da	DANISH	ky	KYRGYZ	sw	SWAHILI
de	GERMAN	la	LATIN	ta	TAMIL
el	GREEK	lb	LUXEMBOURGISH	te	TELUGU
en	ENGLISH	lo	LAOTHIAN	tg	TAJIK
eo	ESPERANTO	lt	LITHUANIAN	th	THAI
es	SPANISH	lv	LATVIAN	tk	TURKMEN
et	ESTONIAN	mg	MALAGASY	tl	TAGALOG
eu	BASQUE	mi	MAORI	tr	TURKISH
fa	PERSIAN	mk	MACEDONIAN	tt	TATAR
fi	FINNISH	ml	MALAYALAM	ug	UIGHUR
fr	FRENCH	mn	MONGOLIAN	uk	UKRAINIAN
fy	FRISIAN	mr	MARATHI	ur	URDU
ga	IRISH	ms	MALAY	uz	UZBEK
gd	SCOTS_GAELIC	mt	MALTESE	vi	VIETNAMESE
gl	GALICIAN	my	BURMESE	wo	WOLOF
gu	GUJARATI	ne	NEPALI	xh	XHOSA
ha	HAUSA	nl	DUTCH	yi	YIDDISH
haw	HAWAIIAN	no	NORWEGIAN	yo	YORUBA
he	HEBREW	ny	NYANJA	zh	Chinese
hi	HINDI	or	ORIYA	zu	ZULU
hmn	HMONG	pa	PUNJABI		
hr	CROATIAN	pl	POLISH		

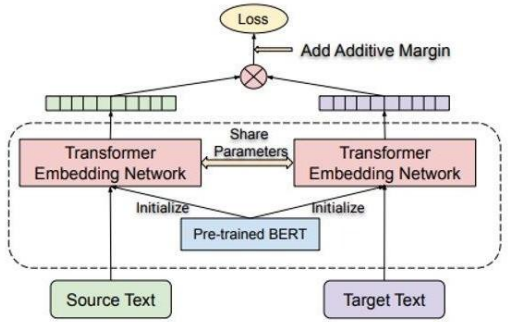


Figure 1: Dual encoder model with BERT based encoding modules.

The most scalable, accurate and fastest arsenal of NLP transformers in history

John Snow LABS

Apache Spark

Core NLP Transformer Models

CamemBertEmbeddings	GloveEmbeddings	Word2VecEmbeddings	Doc2VecEmbeddings
DeBertaEmbeddings	LongformerEmbeddings	BertEmbeddings	XlmRoBertaEmbeddings
DeBertaForQuestionAnswering	LongformerForTokenClassification	BertForTokenClassification	XlmRoBertaForTokenClassification
GPT2Transformer	LongformerForSequenceClassification	BertForSequenceClassification	XlmRoBertaForSequenceClassification
MarianTransformer	LongformerForQuestionAnswering	BertForQuestionAnswering	XlmRoBertaForQuestionAnswering
T5Transformer	UniversalSentenceEncoder	BertSentenceEmbeddings	XlmRoBertaSentenceEmbeddings
RoBertaEmbeddings	XlnetEmbeddings	DistilBertEmbeddings	AlbertEmbeddings
RoBertaForTokenClassification	XlnetForTokenClassification	DistilBertForTokenClassification	AlbertForTokenClassification
RoBertaForSequenceClassification	XlnetForSequenceClassification	DistilBertForSequenceClassification	AlbertForSequenceClassification
RoBertaForQuestionAnswering	ElmoEmbeddings	DistilBertForQuestionAnswering	AlbertForQuestionAnswering

- Transformer for Sequence to Sequence
- Transformer For Token Classification
- Transformer For Sequence Classification
- Transformer For Question Answering
- Transformer Sentence Embeddings
- Transformer Word Embeddings
- Non Transformer Embeddings

Subflavored NLP Transformer Extensions

SciBertEmbeddings	ElectraForQuestionAnswering	MurilBertForQuestionAnswering	MurilBertEmbeddings	LabseSentenceEmbeddings
IndoBertForQuestionAnswering	ElectraSentenceEmbeddings	MurilBertSentenceEmbeddings	SciBertForQuestionAnswering	SapBertForQuestionAnswering
ElectraEmbeddings	BioBertForQuestionAnswering	CovidBertForQuestionAnswering	MiniLmForQuestionAnswering	MacBertForQuestionAnswering
BioFormerForQuestionAnswering	BioBertSentenceEmbeddings	CovidDistilBertForQuestionAnswering	LinkBertForQuestionAnswering	BioBertEmbeddings

Search them all in the Modelshub
https://nlp.johnsnowlabs.com/models

Christian Kasim Loan
Lead Data Scientist
christian@johnsnowlabs.com

Transformers & Embeddings

BERT is a bi-directional transformer for pre-training over a lot of unlabeled textual data to learn a language representation that can be used to fine-tune for specific machine learning tasks. While BERT outperformed the NLP state-of-the-art on several challenging tasks, its performance improvement could be attributed to the bidirectional transformer, novel pre-training tasks of Masked Language Model and Next Structure Prediction along with a lot of data and Google's compute power. It is an Auto Encoder Language Model.

XLNet is a large bidirectional transformer that uses improved training methodology, larger data and more computational power to achieve better than BERT prediction metrics on 20 language tasks.
To improve the training, XLNet introduces permutation language modeling, where all tokens are predicted but in random order. This is in contrast to BERT's masked language model where only the masked (15%) tokens are predicted. It is an Autoregressive Language Model.

Albert is Google's new "ALBERT" language model and achieved state-of-the-art results on three popular benchmark tests for natural language understanding (NLU): GLUE, RACE, and SQuAD 2.0. ALBERT is a "lite" version of Google's 2018 NLU pre training method BERT. Researchers introduced two parameter-reduction techniques in ALBERT to lower memory consumption and increase training speed and the Next Sentence Prediction task is replace by Sentence Order Prediction

USE (Universal Sentence Encoder) is a Transformer-based model for encoding sentences into embedding vectors that specifically target transfer learning to other NLP tasks and outperforms previous word-embedding models on various NLP tasks

Transformers & Embeddings

DistilBert is a compressed version of BERT, which leverages knowledge distillation during the pre-training phase and shows that it is possible to reduce the size of a BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster. It introduces a triple loss combining language modeling, distillation and cosine-distance losses. The smaller, faster and lighter model is cheaper to pre-train and we demonstrate its capabilities for on-device computations in a proof-of-concept experiment and a comparative

on-device

RoBerta is a optimized version of BERT, which has improved hyperparameters and training data size. The findings show that the original BERT was significantly undertrained and with (1) Longer training, bigger batches, (2) removing next sentence prediction objective, (3) training on longer sequences and (4) dynamically changing the masking pattern applied to the training data every previous BERT can be outperformed and new state of the art can be achieved

XlmRoBerta is a multilingual BERT model which significantly outperforms multilingual BERT on a variety of cross-lingual benchmark and large accuracy gains on various multi-lingual benchmarks for 88 languages that appear in the Wiki-100 corpus

Longformer is a Transformer-based model which improves on processing long sequences by introduces a windowed attention mechanism and a linearly scaling self-attention mechanism which scales linearly with sequence length and easily processes documents with thousands or more tokens.

NLP Transformers & Embeddings

- BERT : https://arxiv.org/abs/1810.04805
- XLNet : https://arxiv.org/abs/1906.08237
- Albert : https://arxiv.org/abs/1909.11942
- Elmo : https://arxiv.org/abs/1802.05365
- USE : https://arxiv.org/abs/1803.11175
- DistilBert : https://arxiv.org/abs/1910.01108
- RoBerta : https://arxiv.org/abs/1907.11692
- DeBerta : https://arxiv.org/abs/2006.03654
- XlmRoberta : https://arxiv.org/abs/1911.02116
- Longformer : https://arxiv.org/abs/2004.05150
- Electra : https://arxiv.org/abs/2003.10555
- T5 : https://arxiv.org/abs/1910.10683
- Marian: https://arxiv.org/abs/1804.00344
- GPT2: https://openai.com/blog/better-language-models/
- SpanBERT https://arxiv.org/abs/1907.10529
- CamemBERT https://camembert-model.fr/
- SciBert https://www.aclweb.org/anthology/D19-1371/
- MiniLm https://arxiv.org/abs/2002.10957
- CovidBERT https://arxiv.org/abs/2005.07503
- BioBERT https://arxiv.org/abs/1901.08746
- indoBERT https://arxiv.org/abs/2011.00677
- MuRIL https://arxiv.org/abs/2103.10730
- sapBERT https://github.com/cambridgeltl/sapbert
- BioFormer https://github.com/WGLab/Bioformer
- LinkBERT https://arxiv.org/abs/2203.15827
- MacBERT https://aclanthology.org/2020.findings-emnlp.58
- DOC2Vec : https://arxiv.org/abs/1301.3781
- Word2Vec: https://arxiv.org/abs/1301.3781

Transformers in Spark NLP

https://sparknlp.org/docs/en/transformers

We have extended support for `HuggingFace` 🤗 and `TF Hub` exported models since `3.1.0` to equivalent Spark NLP 🚀 annotators. Starting this release, you can easily use the `saved_model` feature in HuggingFace within a few lines of codes and import any `BERT`, `DistilBERT`, `CamemBERT`, `RoBERTa`, `DeBERTa`, `XLM-RoBERTa`, `Longformer`, `BertForTokenClassification`, `DistilBertForTokenClassification`, `AlbertForTokenClassification`, `RoBertaForTokenClassification`, `DeBertaForTokenClassification`, `XlmRoBertaForTokenClassification`, `XlnetForTokenClassification`, `LongformerForTokenClassification`, `CamemBertForTokenClassification`, `CamemBertForSequenceClassification`, `CamemBertForQuestionAnswering`, `BertForSequenceClassification`, `DistilBertForSequenceClassification`, `AlbertForSequenceClassification`, `RoBertaForSequenceClassification`, `DeBertaForSequenceClassification`, `XlmRoBertaForSequenceClassification`, `XlnetForSequenceClassification`, `LongformerForSequenceClassification`, `AlbertForQuestionAnswering`, `BertForQuestionAnswering`, `DeBertaForQuestionAnswering`, `DistilBertForQuestionAnswering`, `LongformerForQuestionAnswering`, `RoBertaForQuestionAnswering`, `XlmRoBertaForQuestionAnswering`, `TapasForQuestionAnswering`, `Vision Transformers (ViT)`, `HubertForCTC`, `SwinForImageClassification`, and `ConvNextForImageClassification` models to Spark NLP. We will work on the remaining annotators and extend this support to the rest with each release 😊

100+ Languages supported by [Language-agnostic BERT Sentence Embedding (LABSE)] and [XLM-RoBERTa]

Train in **1 Language**, predict in **100+ different languages**

ISO	NAME	ISO	NAME	ISO	NAME
af	AFRIKAANS	ht	HAITIAN_CREOLE	pt	PORTUGUESE
am	AMHARIC	hu	HUNGARIAN	ro	ROMANIAN
ar	ARABIC	hy	ARMENIAN	ru	RUSSIAN
as	ASSAMESE	id	INDONESIAN	rw	KINYARWANDA
az	AZERBAIJANI	ig	IGBO	si	SINHALESE
be	BELARUSIAN	is	ICELANDIC	sk	SLOVAK
bg	BULGARIAN	it	ITALIAN	sl	SLOVENIAN
bn	BENGALI	ja	Japanese	sm	SAMOAN
bo	TIBETAN	jv	JAVANESE	sn	SHONA
bs	BOSNIAN	ka	GEORGIAN	so	SOMALI
ca	CATALAN	kk	KAZAKH	sq	ALBANIAN
ceb	CEBUANO	km	KHMER	sr	SERBIAN
co	CORSICAN	kn	KANNADA	st	SESOTHO
cs	CZECH	ko	KOREAN	su	SUNDANESE
cy	WELSH	ku	KURDISH	sv	SWEDISH
da	DANISH	ky	KYRGYZ	sw	SWAHILI
de	GERMAN	la	LATIN	ta	TAMIL
el	GREEK	lb	LUXEMBOURGISH	te	TELUGU
en	ENGLISH	lo	LAOTHIAN	tg	TAJIK
eo	ESPERANTO	lt	LITHUANIAN	th	THAI
es	SPANISH	lv	LATVIAN	tk	TURKMEN
et	ESTONIAN	mg	MALAGASY	tl	TAGALOG
eu	BASQUE	mi	MAORI	tr	TURKISH
fa	PERSIAN	mk	MACEDONIAN	tt	TATAR
fi	FINNISH	ml	MALAYALAM	ug	UIGHUR
fr	FRENCH	mn	MONGOLIAN	uk	UKRAINIAN
fy	FRISIAN	mr	MARATHI	ur	URDU
ga	IRISH	ms	MALAY	uz	UZBEK
gd	SCOTS_GAELIC	mt	MALTESE	vi	VIETNAMESE
gl	GALICIAN	my	BURMESE	wo	WOLOF
gu	GUJARATI	ne	NEPALI	xh	XHOSA
ha	HAUSA	nl	DUTCH	yi	YIDDISH
haw	HAWAIIAN	no	NORWEGIAN	yo	YORUBA
he	HEBREW	ny	NYANJA	zh	Chinese
hi	HINDI	or	ORIYA	zu	ZULU
hmn	HMONG	pa	PUNJABI		
hr	CROATIAN	pl	POLISH		

```
# Binary Class Classifier, 2 classes
nlu.load('xx.embed_sentence.labse train.sentiment').fit(train_df).predict(test_df)

# Multi Class Classifier, N classes
nlu.load('xx.embed_sentence.labse train.classifier').fit(train_df).predict(test_df)

# Multi Class Classifier with multiple labels example (i.e. Hashtags)
# N classes, where one row can be assigned up to N labels
nlu.load('xx.embed_sentence.labse train.multi_classifier').fit(train_df).predict(test_df)
```

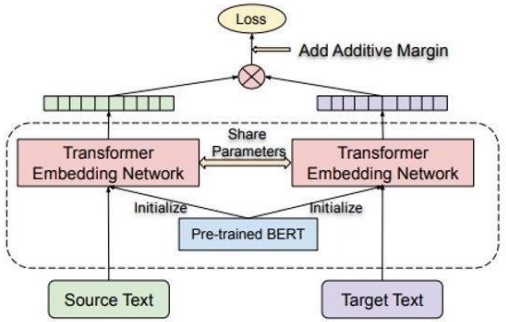
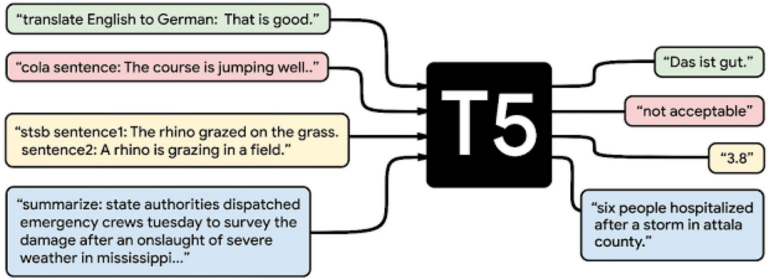


Figure 1: Dual encoder model with BERT based encoding modules.




```
# Closed book Question Answering
nlu.load('en.t5').predict('what is the capital of Germany?') # >>> Berlin
# Open Book Question answering
nlu.load('en.t5').predict('Who is president of Nigeria?') # >>> Muhammadu Buhari


# Open book Question Answering
context   = 'Peters last week was terrible! He had an accident and broke his leg while skiing!'
question1  = 'Why was peters week so bad?'
question2  = 'How did peter broke his leg?'
nlu.load('answer_question').predict(question1 + context) # >>> broke his leg
nlu.load('answer_question').predict(question2 + context) # >>> skiing



# Big T5 model for Summarization, Sentiment, Text Similarity and other SQUAD/GLUE tasks
pipe = nlu.load('t5')
pipe['t5'].settask('summarize')
pipe.predict(long_text)
```

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

1. Text summarization
2. Question answering
3. Translation
4. Sentiment analysis
5. Natural Language inference
6. Coreference resolution
7. Sentence Completion
8. Word sense disambiguation

Google

Every T5 Task with explanation:

Task Name	Explanation
1.CoLA	Classify if a sentence is grammatically correct
2.RTE	Classify whether a statement can be deducted from a sentence
3.MNLI	Classify for a hypothesis and premise whether they contradict or contradict each other or neither of both (3 class).
4.MRPC	Classify whether a pair of sentences is a re-phrasing of each other (semantically equivalent)
5.QNLI	Classify whether the answer to a question can be deducted from an answer candidate.
6.QQP	Classify whether a pair of questions is a re-phrasing of each other (semantically equivalent)
7.SST2	Classify the sentiment of a sentence as positive or negative
8.STSB	Classify the sentiment of a sentence on a scale from 1 to 5 (21 Sentiment classes)
9.CB	Classify for a premise and a hypothesis whether they contradict each other or not (binary).
10.COPA	Classify for a question, premise, and 2 choices which choice the correct choice is (binary).
11.MultiRc	Classify for a question, a paragraph of text, and an answer candidate, if the answer is correct (binary),
12.WiC	Classify for a pair of sentences and a disambigous word if the word has the same meaning in both sentences.
13.WSC/DPR	Predict for an ambiguous pronoun in a sentence what it is referring to.
14.Summarization	Summarize text into a shorter representation.
15.SQuAD	Answer a question for a given context.
16.WMT1.	Translate English to German
17.WMT2.	Translate English to French
18.WMT3.	Translate English to Romanian

Train Transformer Models via Hugging Face or TfHub and scale with Spark NLP

TensorFlow Hub

TF Hub to Spark NLP

Spark NLP	TF Hub Notebooks	Colab
BertEmbeddings	TF Hub in Spark NLP - BERT	Open in Colab
BertSentenceEmbeddings	TF Hub in Spark NLP - BERT Sentence	Open in Colab
AlbertEmbeddings	TF Hub in Spark NLP - ALBERT	Open in Colab

Spark NLP	HuggingFace Notebooks	Colab
BertEmbeddings	HuggingFace in Spark NLP - BERT	Open in Colab
BertSentenceEmbeddings	HuggingFace in Spark NLP - BERT Sentence	Open in Colab
DistilBertEmbeddings	HuggingFace in Spark NLP - DistilBERT	Open in Colab
CamemBertEmbeddings	HuggingFace in Spark NLP - CamemBERT	Open in Colab
RoBertaEmbeddings	HuggingFace in Spark NLP - RoBERTa	Open in Colab
DeBertaEmbeddings	HuggingFace in Spark NLP - DeBERTa	Open in Colab
XlmRoBertaEmbeddings	HuggingFace in Spark NLP - XLM-RoBERTa	Open in Colab
AlbertEmbeddings	HuggingFace in Spark NLP - ALBERT	Open in Colab
XlnetEmbeddings	HuggingFace in Spark NLP - XLNet	Open in Colab
LongformerEmbeddings	HuggingFace in Spark NLP - Longformer	Open in Colab
BertForTokenClassification	HuggingFace in Spark NLP - BertForTokenClassification	Open in Colab
DistilBertForTokenClassification	HuggingFace in Spark NLP - DistilBertForTokenClassification	Open in Colab
AlbertForTokenClassification	HuggingFace in Spark NLP - AlbertForTokenClassification	Open in Colab
RoBertaForTokenClassification	HuggingFace in Spark NLP - RoBertaForTokenClassification	Open in Colab
XlmRoBertaForTokenClassification	HuggingFace in Spark NLP - XlmRoBertaForTokenClassification	Open in Colab
BertForSequenceClassification	HuggingFace in Spark NLP - BertForSequenceClassification	Open in Colab
DistilBertForSequenceClassification	HuggingFace in Spark NLP - DistilBertForSequenceClassification	Open in Colab
AlbertForSequenceClassification	HuggingFace in Spark NLP - AlbertForSequenceClassification	Open in Colab
RoBertaForSequenceClassification	HuggingFace in Spark NLP - RoBertaForSequenceClassification	Open in Colab
XlmRoBertaForSequenceClassification	HuggingFace in Spark NLP - XlmRoBertaForSequenceClassification	Open in Colab
XlnetForSequenceClassification	HuggingFace in Spark NLP - XlnetForSequenceClassification	Open in Colab
LongformerForSequenceClassification	HuggingFace in Spark NLP - LongformerForSequenceClassification	Open in Colab
AlbertForQuestionAnswering	HuggingFace in Spark NLP - AlbertForQuestionAnswering	Open in Colab
BertForQuestionAnswering	HuggingFace in Spark NLP - BertForQuestionAnswering	Open in Colab
DeBertaForQuestionAnswering	HuggingFace in Spark NLP - DeBertaForQuestionAnswering	Open in Colab
DistilBertForQuestionAnswering	HuggingFace in Spark NLP - DistilBertForQuestionAnswering	Open in Colab
LongformerForQuestionAnswering	HuggingFace in Spark NLP - LongformerForQuestionAnswering	Open in Colab
RoBertaForQuestionAnswering	HuggingFace in Spark NLP - RoBertaForQuestionAnswering	Open in Colab
XlmRobertaForQuestionAnswering	HuggingFace in Spark NLP - XlmRobertaForQuestionAnswering	Open in Colab

https://nlp.johnsnowlabs.com/docs/en/transformers#example-notebooks

Session 2 (Day 1) - Coding Time

- Notebook 2 Text Preprocessing and composing Pipelines
- Notebook 3.1 - Pretrained Models Overview

Spark NLP
for Data Scientists

John Snow LABS

Session 3 (Day 1)

- Usage and overview of the 10000+ pretrained models for 300+ languages
- Overview NER, Text Classification

Spark NLP
for Data Scientists

Not Secure | nlpprogress.com/eng...

CoNLL 2003 (English)

The CoNLL 2003 NER task consists of newswire text from the Reuters RCV1 corpus tagged with four different entity types (PER, LOC, ORG, MISC). Models are evaluated based on span-based F1 on the test set. ♦ used both the train and development splits for training.

Model	F1	Paper / Source	Code
CNN Large + fine-tune (Baevski et al., 2019)	93.5	Cloze-driven Pretraining of Self-attention Networks	
RNN-CRF+Flair	93.47	Improved Differentiable Architecture Search for Language Modeling and Named Entity Recognition	
LSTM-CRF+ELMo+BERT+Flair	93.38	Neural Architectures for Nested NER through Linearization	Official
Flair embeddings (Akbik et al., 2018)♦	93.09	Contextual String Embeddings for Sequence Labeling	Flair framework
BERT Large (Devlin et al., 2018)	92.8	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	
CVT + Multi-Task (Clark et al., 2018)	92.61	Semi-Supervised Sequence Modeling with Cross-View Training	Official
BERT Base (Devlin et al., 2018)	92.4	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	
BiLSTM-CRF+ELMo (Peters et al., 2018)	92.22	Deep contextualized word representations	AllenNLP Project AllenNLP GitHub
Peters et al. (2017) ♦	91.93	Semi-supervised sequence tagging with bidirectional language models	
CRF + AutoEncoder (Wu et al., 2018)	91.87	Evaluating the Utility of Hand-crafted Features in Sequence Labelling	Official
Bi-LSTM-CRF + Lexical Features (Ghaddar and Langlais 2018)	91.73	Robust Lexical Features for Improved Neural Network Named-Entity Recognition	Official
BiLSTM-CRF + IntNet (Xin et al., 2018)	91.64	Learning Better Internal Structure of Words for Sequence Labeling	
Chiu and Nichols (2016) ♦	91.62	Named entity recognition with bidirectional LSTM-CNNs	

NER-DL in Spark NLP

SYSTEM	YEAR	LANGUAGE	ACCURACY
Spark NLP v2.4	2020	Python/Scala/Java/R	**93.3 (test F1) - 95.9 (dev F1)**
Spark NLP v2.x	2019	Python/Scala/Java/R	**93**
Spark NLP v1.x	2018	Python/Scala/Java/R	92
spaCy v2.x	2017	Python/Cython	92.6
spaCy v1.x	2015	Python/Cython	91.8
ClearNLP	2015	Java	91.7
CoreNLP	2015	Java	89.6
MATE	2015	Java	92.5
Turbo	2015	C++	92.4

The best NER score in production

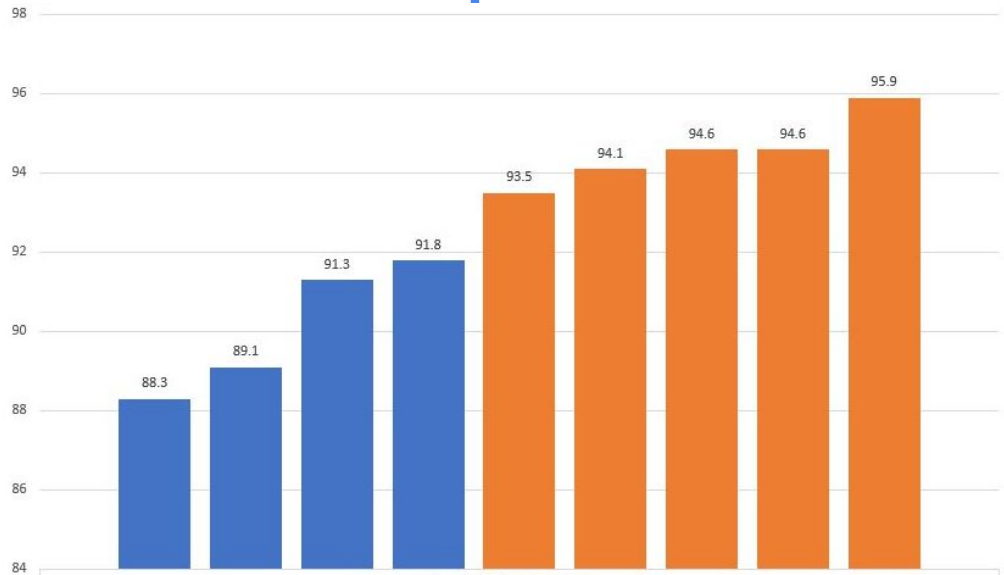
93.3 %
Test Set

Bert

NerDLApproach

John Snow LABS

NER-DL in Spark NLP



	CoNLL 2003 - F1 Dev
spaCy 2.2	88.3
spaCy 2.2 + en_trf_bertbaseuncased_lg	89.1
spaCy 2.2 + en_core_web_md	91.3
spaCy 2.2 + en_core_web_lg	91.8
Spark NLP + GloVe 100d	93.5
Spark NLP + GloVe 300d	94.1
Spark NLP + ELMO	94.6
Spark NLP + BERT Base Cased	94.6
Spark NLP + BERT Large Cased	95.9

NER Systems

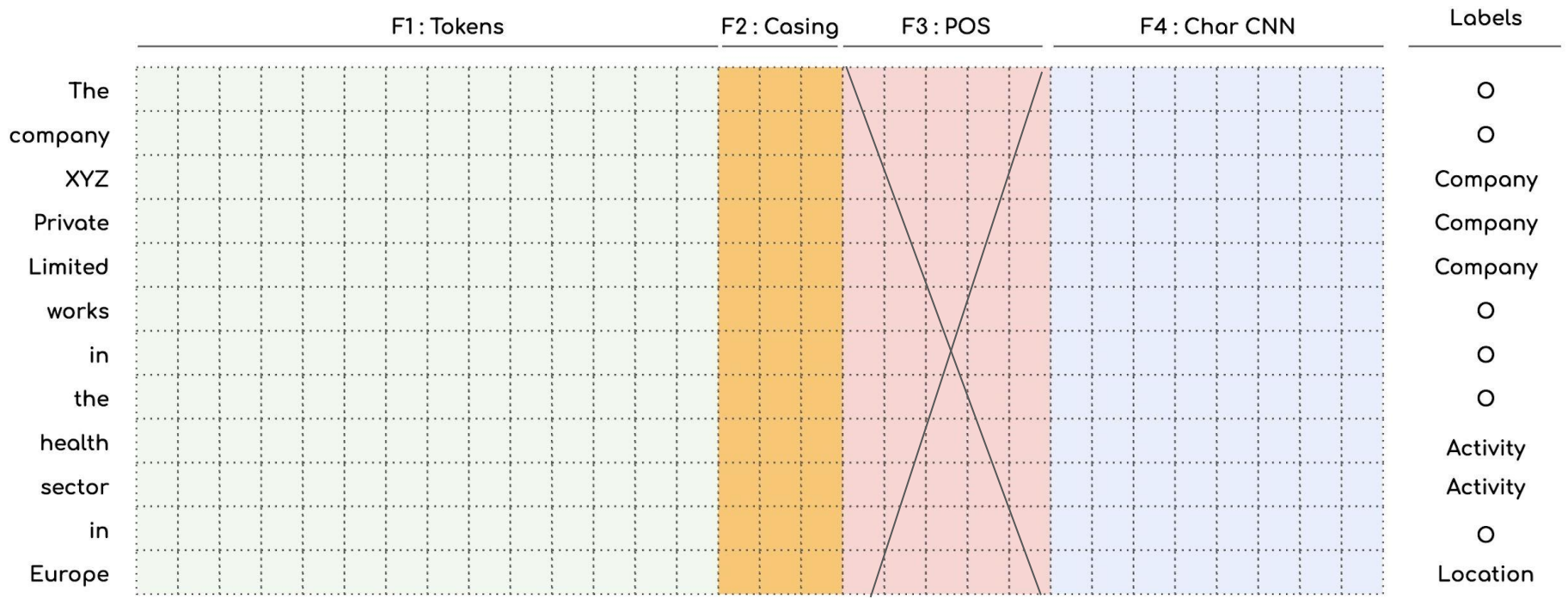
Feature-engineered machine learning systems	Dict	SP	DU	EN	GE
Carreras et al. (2002) binary AdaBoost classifiers	Yes	81.39	77.05	-	-
Malouf (2002) - Maximum Entropy (ME) + features	Yes	73.66	68.08	-	-
Li et al. (2005) SVM with class weights	Yes	-	-	88.3	-
Passos et al. (2014) CRF	Yes	-	-	90.90	-
Ando and Zhang (2005a) Semi-supervised state of the art	No	-	-	89.31	75.27
Agerri and Rigau (2016)	Yes	**84.16**	**85.04**	**91.36**	**76.42**
Feature-inferring neural network word models					
Collobert et al. (2011) Vanilla NN +SLL / Conv-CRF	No	-	-	81.47	-
Huang et al. (2015) Bi-LSTM+CRF	No	-	-	84.26	-
Yan et al. (2016) Win-BiLSTM (English), FF (German) (Many fets)	Yes	-	-	88.91	**76.12**
Collobert et al. (2011) Conv-CRF (SENNA+Gazetteer)	Yes	-	-	89.59	-
Huang et al. (2015) Bi-LSTM+CRF+ (SENNA+Gazetteer)	Yes	-	-	**90.10**	-
Feature-inferring neural network character models					
Gillick et al. (2015) – BTS	No	**82.95**	**82.84**	**86.50**	**76.22**
Kuru et al. (2016) CharNER	No	82.18	79.36	84.52	70.12
Feature-inferring neural network word + character models					
Yang et al. (2017)	Yes	85.77	**85.19**	91.26	-
Luo (2015)	Yes	-	-	91.20	-
Chiu and Nichols (2015)	Yes	-	-	**91.62**	-
Ma and Hovy (2016)	No	-	-	91.21	-
Santos and Guimaraes (2015)	No	82.21	-	-	-
Lample et al. (2016)	No	85.75	81.74	90.94	**78.76**
Bharadwaj et al. (2016)	Yes	**85.81**	-	-	-
Dernoncourt et al. (2017)	No	-	-	90.5	-
Feature-inferring neural network word + character + affix models					
Re-implementation of Lample et al. (2016) (100 Epochs)	No	85.34	85.27	90.24	78.44
Yadav et al. (2018)(100 Epochs)	No	86.92	87.50	90.69	78.56
Yadav et al. (2018) (150 Epochs)	No	**87.26**	**87.54**	90.86	**79.01**

1. Classical Approaches (rule based)
2. ML Approaches
   - Multi-class classification
   - Conditional Random Field (CRF)
3. DL Approaches
   - Bidirectional LSTM-CRF
   - Bidirectional LSTM-CNNs
   - Bidirectional LSTM-CNNS-CRF
   - Pre-trained language models (Bert, Elmo)
4. Hybrid Approaches (DL + ML)

John Snow LABS

NER-DL in Spark NLP

Char-CNN-BiLSTM



NER-DL in Spark NLP

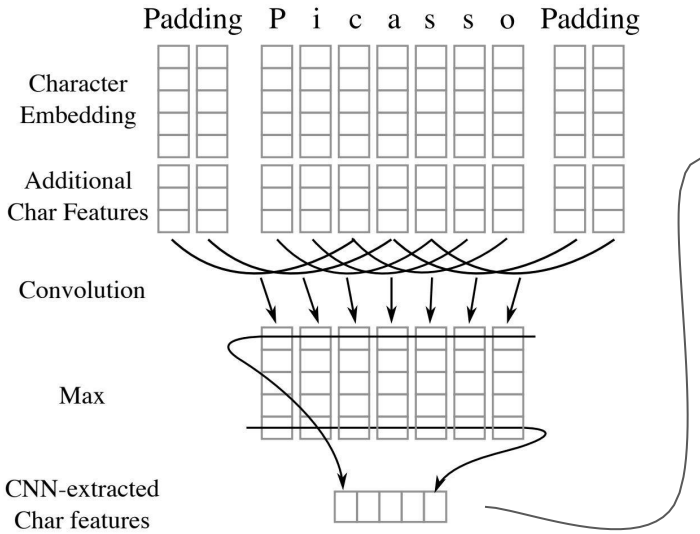


Figure 2: The convolutional neural network extracts character features from each word. The character embedding and (optionally) the character type feature vector are computed through lookup tables. Then, they are concatenated and passed into the CNN.

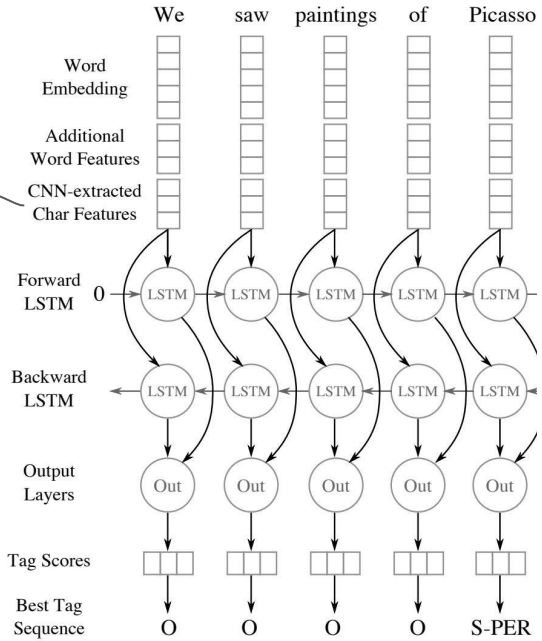


Figure 1: The (unrolled) BLSTM for tagging named entities. Multiple tables look up word-level feature vectors. The CNN (Figure 2) extracts a fixed length feature vector from character-level features. For each word, these vectors are concatenated and fed to the BLSTM network and then to the output layers (Figure 3).

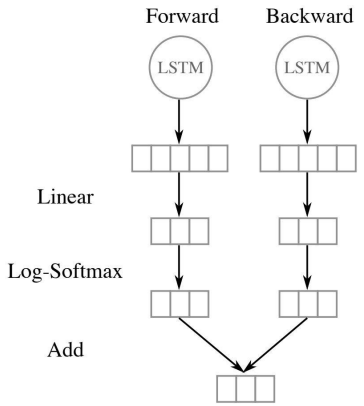


Figure 3: The output layers ("Out" in Figure 1) decode output into a score for each tag category.

Char-CNN-BiLSTM

John Snow LABS

NER-DL in Spark NLP

CoNNL2003 format

All data files contain one word per line with empty lines representing sentence boundaries. At the end of each line there is a tag which states whether the current word is inside a named entity or not. The tag also encodes the type of named entity. Here is an example sentence:

U.N.	NNP	I-NP	I-ORG
official	NN	I-NP	O
Ekeus	NNP	I-NP	I-PER
heads	VBZ	I-VP	O
for	IN	I-PP	O
Baghdad	NNP	I-NP	I-LOC
.	.	O	O

*Each line contains four fields: the word, its part-of-speech tag, its chunk tag and its named entity tag.

BIO schema

```
John      B-PER
Smith     I-PER
lives     O
in        O
New       B-LOC
York      I-LOC
```

John Smith ⇒ PERSON
New York ⇒ LOCATION

* **CoNLL**: Conference on Computational Natural Language Learning

John Snow LABS

Session 3 (Day 1) - Coding Time

- Notebook 3 SparkNLP_Pretrained_Models
- Notebook 4.2 Transformers for Token Classification
- Notebook 4.4 ZeroShot NER
- Notebook 5.2 Transformers for Sequence Classification

Spark NLP
for Data Scientists

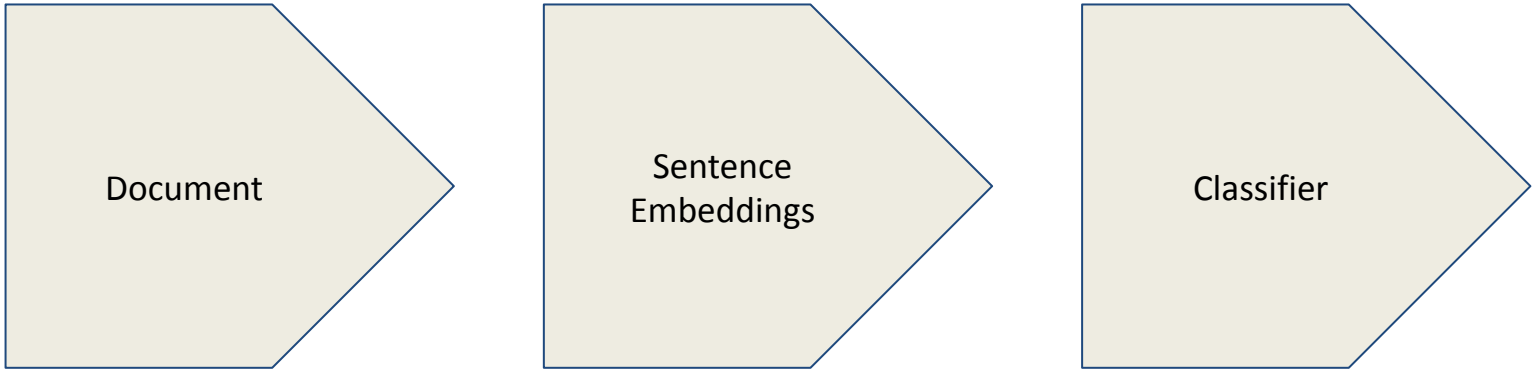
John Snow LABS

Session 4 (Day 1)

- NER Training
- Text Classification Training

Spark NLP
for Data Scientists

Spark NLP Classifiers



SentimentDL, ClassifierDL, and MultiClassifierDL

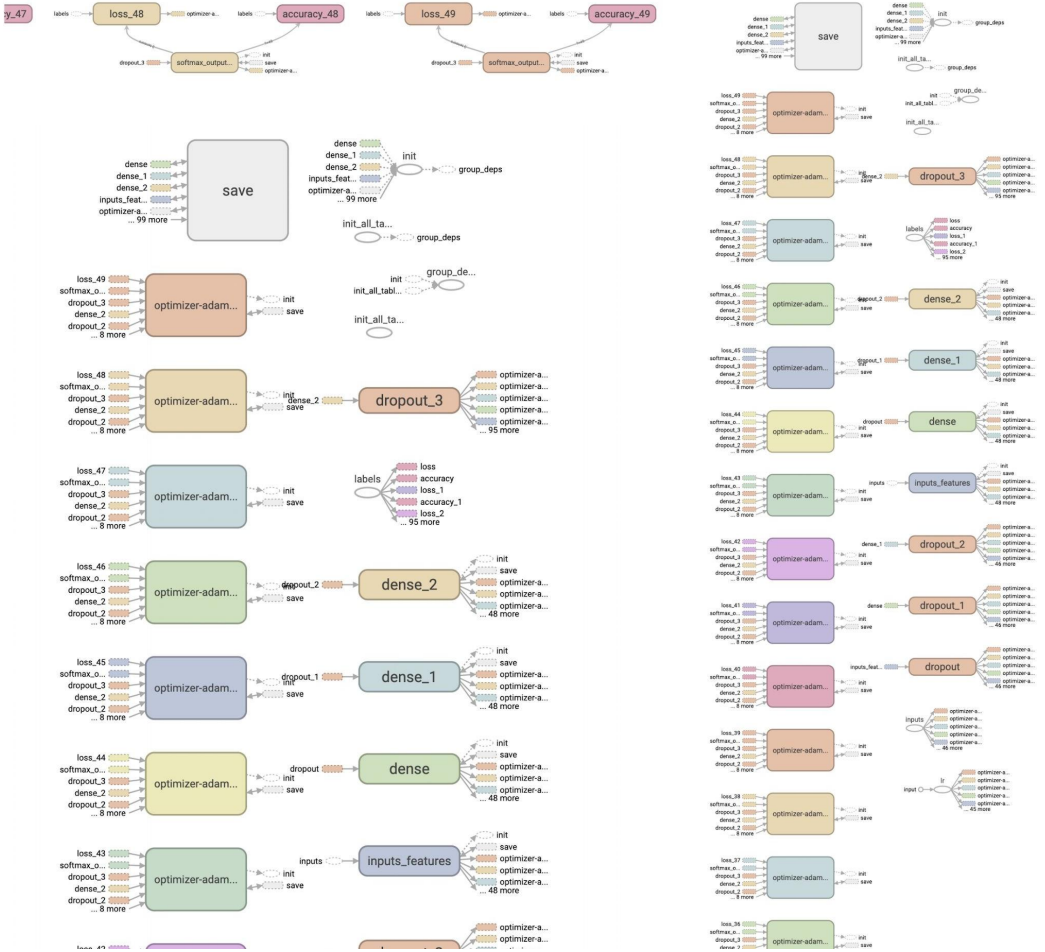
- BERT
- Small BERT
- BioBERT
- CovidBERT
- LaBSE
- ALBERT
- ELECTRA
- XLNet
- ELMO
- Universal Sentence Encoder
- GloVe
- Mpnet
- E5

- 2 classes (positive/negative)
- 3 classes (0, 1, 2)
- 4 classes (Sports, Business, etc.)
- 5 classes (1.0, 2.0, 3.0, 4.0, 5.0)
- ... 100 classes!

- 100 dimensions
- 200 dimensions
- 128 dimensions
- 256 dimensions
- 300 dimensions
- 512 dimensions
- 768 dimensions
- 1024 dimensions

- tfhub_ues
- tfhub_use_lg
- glove_6B_100
- glove_6B_300
- glove_840B_300
- bert_base_cased
- bert_base_uncased
- bert_large_cased
- bert_large_uncased
- bert_multi_uncased
- electra_small_uncased
- elmo
- ... 100+ Word & Sentence models

Classifier DL Tensorflow Architecture



Session 4 (Day 1) - Coding Time

- Notebook 4: NERDL Training
- Notebook 5: Classification Training, classifierDL, multi-classifier

Spark NLP
for Data Scientists

John Snow LABS

Coding ...

Open **4. NERDL Training** notebook in **Colab**

(click on Colab icon or open in a new tab)

https://github.com/JohnSnowLabs/spark-nlp-workshop/blob/master/open-source-nlp/04.0.NERDL_Training.ipynb

Test set evaluation

In [ ]:
```
import pyspark.sql.functions as F

predictions.select(F.explode(F.arrays_zip('token.result','label.result','ner.result')).alias("cols")) \
.select(F.expr("cols['0']").alias("token"),
        F.expr("cols['1']").alias("ground_truth"),
        F.expr("cols['2']").alias("prediction")).show(truncate=False)
```

```
+--------------+------------+----------+
|token         |ground_truth|prediction|
+--------------+------------+----------+
|CRICKET       |O           |O         |
|-             |O           |O         |
|LEICESTERSHIRE|B-ORG       |B-ORG     |
|TAKE          |O           |O         |
|OVER          |O           |O         |
|AT            |O           |O         |
|TOP           |O           |O         |
|AFTER         |O           |O         |
|INNINGS       |O           |O         |
|VICTORY       |O           |O         |
|.             |O           |O         |
|LONDON        |B-LOC       |B-LOC     |
|1996-08-30    |O           |O         |
|West          |B-MISC      |B-MISC    |
|Indian        |I-MISC      |I-MISC    |
|all-rounder   |O           |O         |
|Phil          |B-PER       |B-PER     |
|Simmons       |I-PER       |I-PER     |
|took          |O           |O         |
|four          |O           |O         |
+--------------+------------+----------+
only showing top 20 rows
```

In [ ]:
```
from sklearn.metrics import classification_report

preds_df = predictions.select(F.explode(F.arrays_zip('token.result','label.result','ner.result')).alias("cols")) \
.select(F.expr("cols['0']").alias("token"),
        F.expr("cols['1']").alias("ground_truth"),
        F.expr("cols['2']").alias("prediction")).toPandas()


print (classification_report(preds_df['ground_truth'], preds_df['prediction']))
```

```
              precision    recall  f1-score   support

       B-LOC       0.88      0.93      0.90      1837
      B-MISC       0.80      0.82      0.81       922
       B-ORG       0.92      0.73      0.81      1341
       B-PER       0.94      0.95      0.95      1842
```

John Snow LABS

End of Day 1 - see you tomorrow!
Same time, same place :)

Spark NLP
for Data Scientists

John Snow LABS

Welcome - We have a lot of things ahead of us

Day-2	50 min	- Summarization - Question-Answering
	10 min	Break
	50 min	- Automatic Speech Recognition - Image Classification - Neural Machine Translation
	10 min	Break
	50 min	- Splitting documents into meaningful chunks - Ranking retrieved documents - RAG
	10 min	Break
	50 min	- OpenIAI in Spark NLP - Import Transformers into Spark NLP

Session 1 (Day 2)

- Question Answering
- Summarization
- Transformers-based models

Spark NLP
for Data Scientists

John Snow LABS

TAPAS for Table Classification

[Notebook 17 Table Question Answering with TAPAS]

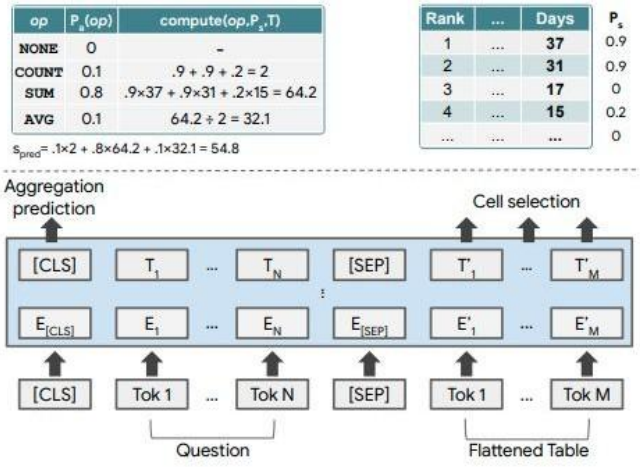


Figure 1: TAPAS model (bottom) with example model outputs for the question: *"Total number of days for the top two"*. Cell prediction (top right) is given for the selected column's table cells in bold (zero for others) along with aggregation prediction (top left).

[TAPAS: Weakly Supervised Table Parsing via Pre-training]

TAPAS for Table Question Answering using Spark NLP 🚀

TAPAS is a Zero-shot Question Answering architecture, based on Bert, to carry out Table Understanding. By asking a question on natural language using these models, you can retrieve the content of the cell or cells which best answer to those questions.

Table

Rank	Name	No. of reigns	Combined days
1	Lou Thesz	3	3,749
2	Ric Flair	8	3,103
3	Harley Race	7	1,799
4	Dory Funk Jr.	1	1,563
5	Dan Severn	2	1,559
6	Gene Kiniski	1	1,131

Example questions

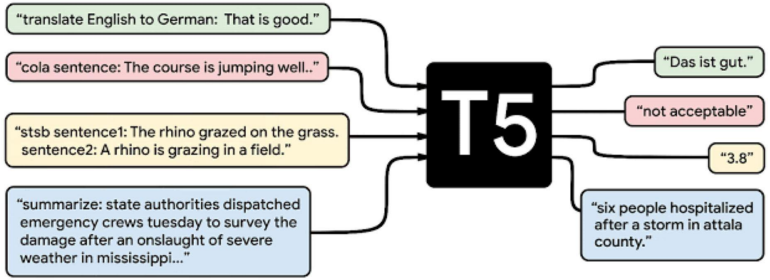
#	Question	Answer	Example Type
1	*Which wrestler had the most number of reigns?*	Ric Flair	Cell selection
2	*Average time as champion for top 2 wrestlers?*	AVG(3749,3103)=3426	Scalar answer
3	*How many world champions are there with only one reign?*	COUNT(Dory Funk Jr., Gene Kiniski)=2	Ambiguous answer
4	*What is the number of reigns for* Harley Race?	7	Ambiguous answer
5	*Which of the following wrestlers were ranked in the bottom 3?*	{Dory Funk Jr., Dan Severn, Gene Kiniski}	Cell selection
	Out of these, who had more than one reign?	Dan Severn	Cell selection

TAPAS models have been trained using a combination of three datasets:

- SQA, Sequential Question Answering by Microsoft (it was not trained to return aggregation operations as SUM, COUNT, etc - see below)
- WTQ, Wiki Table Questions by Stanford University (with aggregation operations)
- Wiki SQL, by Salesforce, also with aggregation operations.

Google

[Table Classifiers Overview]




```
# Closed book Question Answering
nlu.load('en.t5').predict('what is the capital of Germany?') # >>> Berlin
# Open Book Question answering
nlu.load('en.t5').predict('Who is president of Nigeria?') # >>> Muhammadu Buhari


# Open book Question Answering
context   = 'Peters last week was terrible! He had an accident and broke his leg while skiing!'
question1  = 'Why was peters week so bad?'
question2  = 'How did peter broke his leg?'
nlu.load('answer_question').predict(question1 + context) # >>> broke his leg
nlu.load('answer_question').predict(question2 + context) # >>> skiing



# Big T5 model for Summarization, Sentiment, Text Similarity and other SQUAD/GLUE tasks
pipe = nlu.load('t5')
pipe['t5'].settask('summarize')
pipe.predict(long_text)
```

[Notebook 10 Question Answering and Summarization and more with T5]

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

1. Text summarization
2. Question answering
3. Translation
4. Generate SQL from natural language text
5. Text style transfer
6. Sentiment analysis
7. Natural Language inference
8. Coreference resolution
9. Sentence Completion
10. Word sense disambiguation

Every T5 Task with explanation:

Task Name	Explanation
1.CoLA	Classify if a sentence is grammatically correct
2.RTE	Classify whether a statement can be deducted from a sentence
3.MNLI	Classify for a hypothesis and premise whether they contradict or contradict each other or neither of both (3 class).
4.MRPC	Classify whether a pair of sentences is a re-phrasing of each other (semantically equivalent)
5.QNLI	Classify whether the answer to a question can be deducted from an answer candidate.
6.QQP	Classify whether a pair of questions is a re-phrasing of each other (semantically equivalent)
7.SST2	Classify the sentiment of a sentence as positive or negative
8.STSB	Classify the sentiment of a sentence on a scale from 1 to 5 (21 Sentiment classes)
9.CB	Classify for a premise and a hypothesis whether they contradict each other or not (binary).
10.COPA	Classify for a question, premise, and 2 choices which choice the correct choice is (binary).
11.MultiRc	Classify for a question, a paragraph of text, and an answer candidate, if the answer is correct (binary),
12.WiC	Classify for a pair of sentences and a disambigous word if the word has the same meaning in both sentences.
13.WSC/DPR	Predict for an ambiguous pronoun in a sentence what it is referring to.
14.Summarization	Summarize text into a shorter representation.
15.SQuAD	Answer a question for a given context.
16.WMT1.	Translate English to German
17.WMT2.	Translate English to French
18.WMT3.	Translate English to Romanian

Session 1 (Day 2) - Coding Time

- Notebook 15 - Question Answering
- Notebook 18 - Summarization
- Notebook 13 - Q&A and Summarization with T5
- Notebook 15.1 - Table Question Answering

Spark NLP
for Data Scientists

John Snow LABS

Session 2 (Day 2)

- Automatic Speech Recognition (ASR)
- Image Classification
- Neural Machine Translation (NMT)

Spark NLP
for Data Scientists

John Snow LABS

Image Classification with ViT

[Notebook 18 VIT for Image Classification]

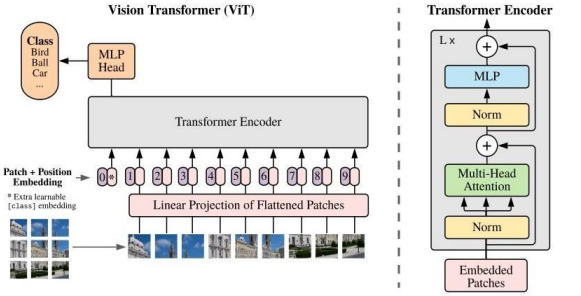
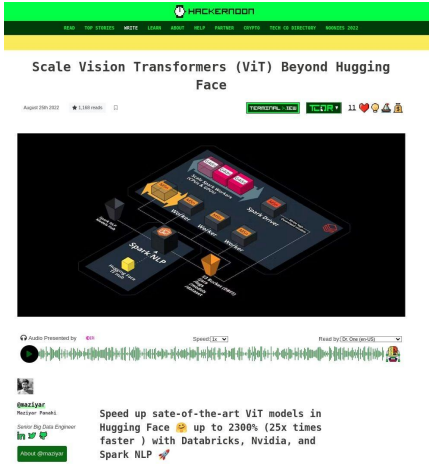
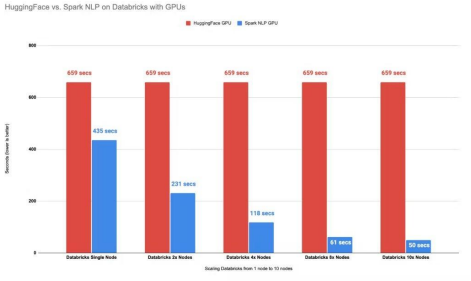


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

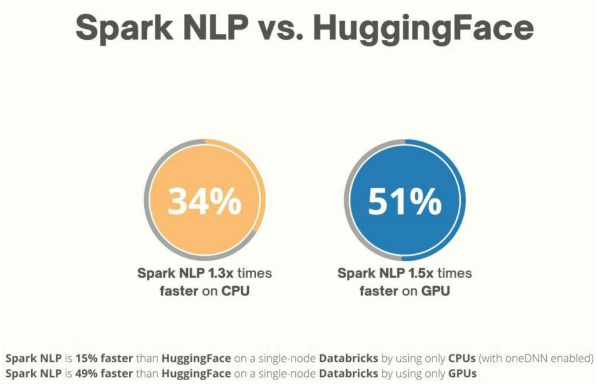


[Scale Vision Transformers (ViT) Beyond Hugging Face]



Spark NLP is **1200% faster** than Hugging Face with **10x Nodes**

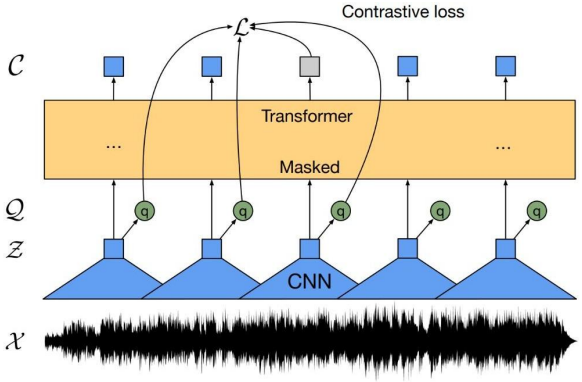
[VIT: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale]



[Over 200 VIT Models in various languages and sizes]

Speech to Text

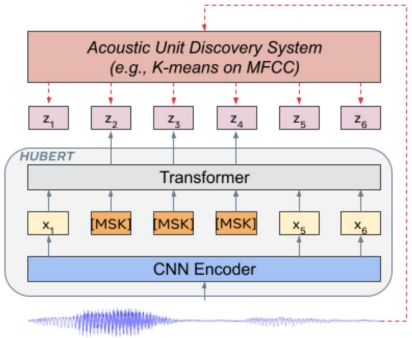
Wav2Vec



Meta

[wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations]

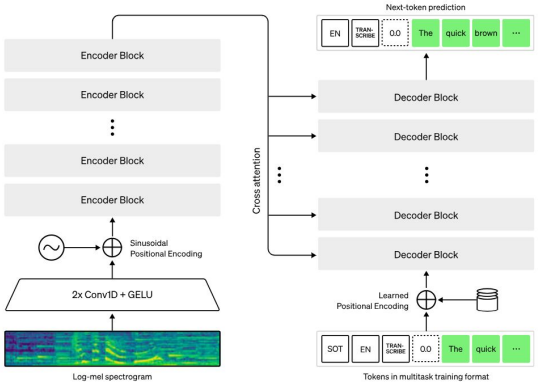
HuBERT



Meta

[HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units Representations]

Whisper



OpenAI

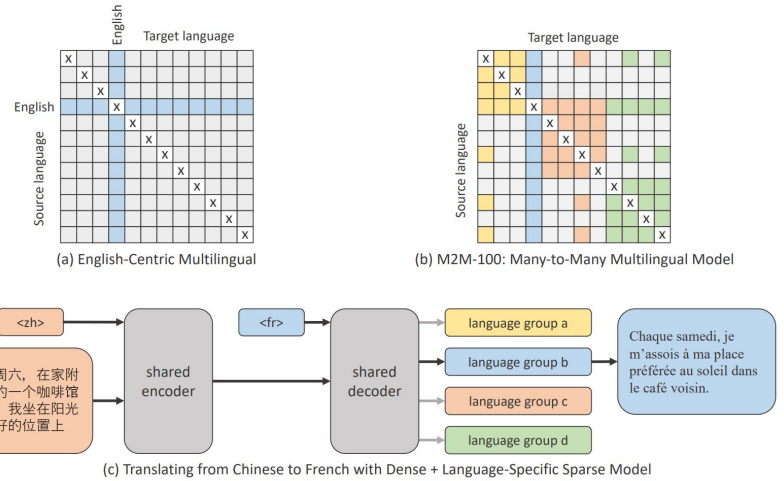
[Robust Speech Recognition via Large-Scale Weak Supervision Representations]

Neural Machine Translation

Translate between **200+ Languages**

[With Marian: Fast Neural Machine Translation in C++]

[Beyond English-Centric Multilingual Machine Translation with M2M-100]



Session 2 (Day 2) - Coding Time

- Notebook 16 Image Classification
- Notebook 17 Automatic Speech Recognition
- Notebook 20 Neural Machine Translation

Spark NLP
for Data Scientists

John Snow LABS

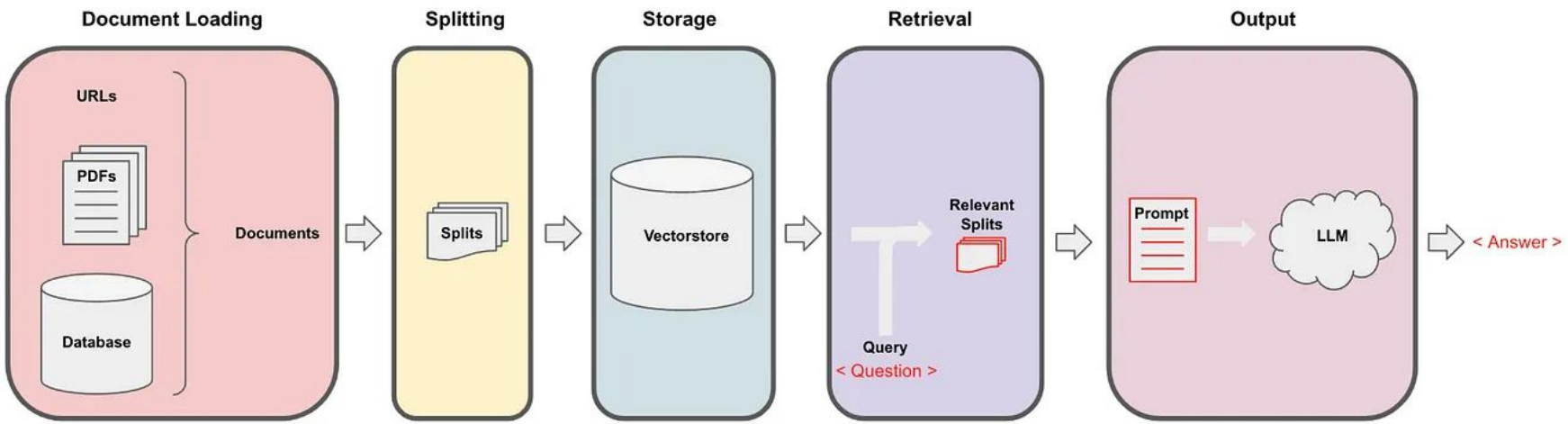
Session 3 (Day 2)

- Document Splitters
- Document Similarity Ranker

Spark NLP
for Data Scientists

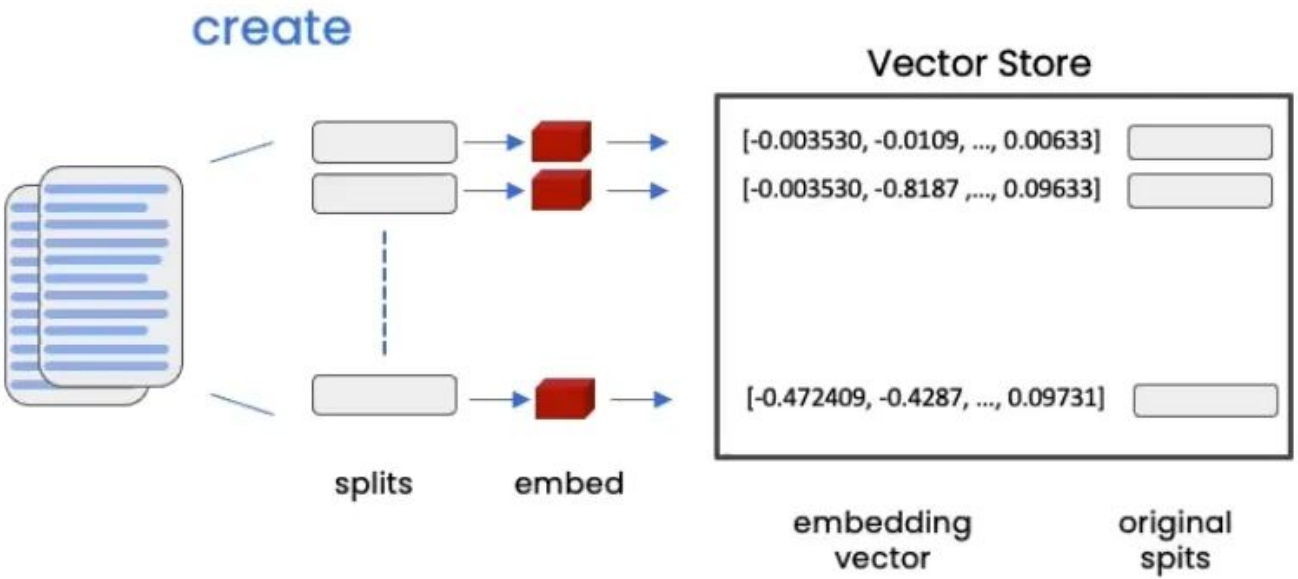
John Snow LABS

Document Splitters



picture

Document Splitters



picture

Session 3 (Day 2) - Coding Time

- 21.0.Document_Splitters.ipynb
- 21.1.Document_Similarity_Ranker.ipynb

Spark NLP
for Data Scientists

John Snow LABS

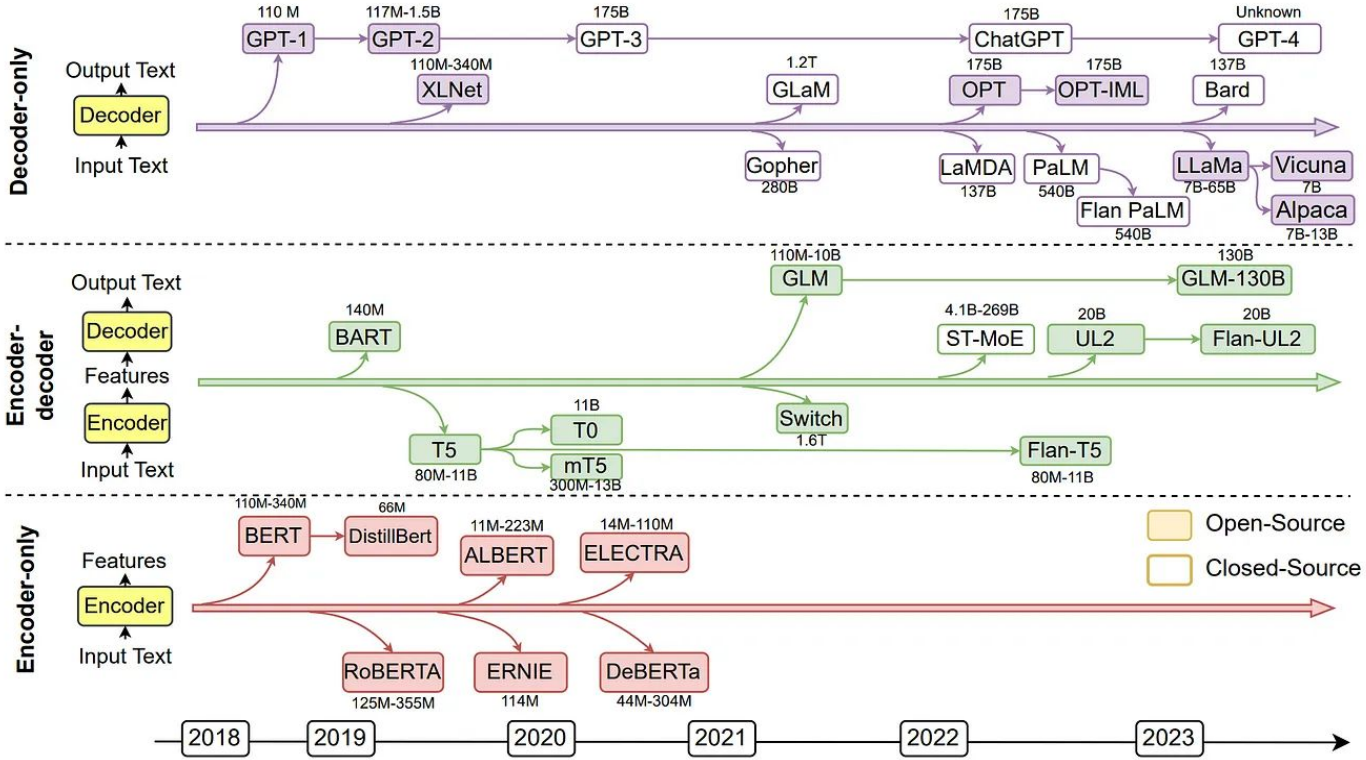
Session 4 (Day 2)

- OpenIA in Spark NLP
- Llama2 in Spark NLP
- Import Transformers into SparkNLP

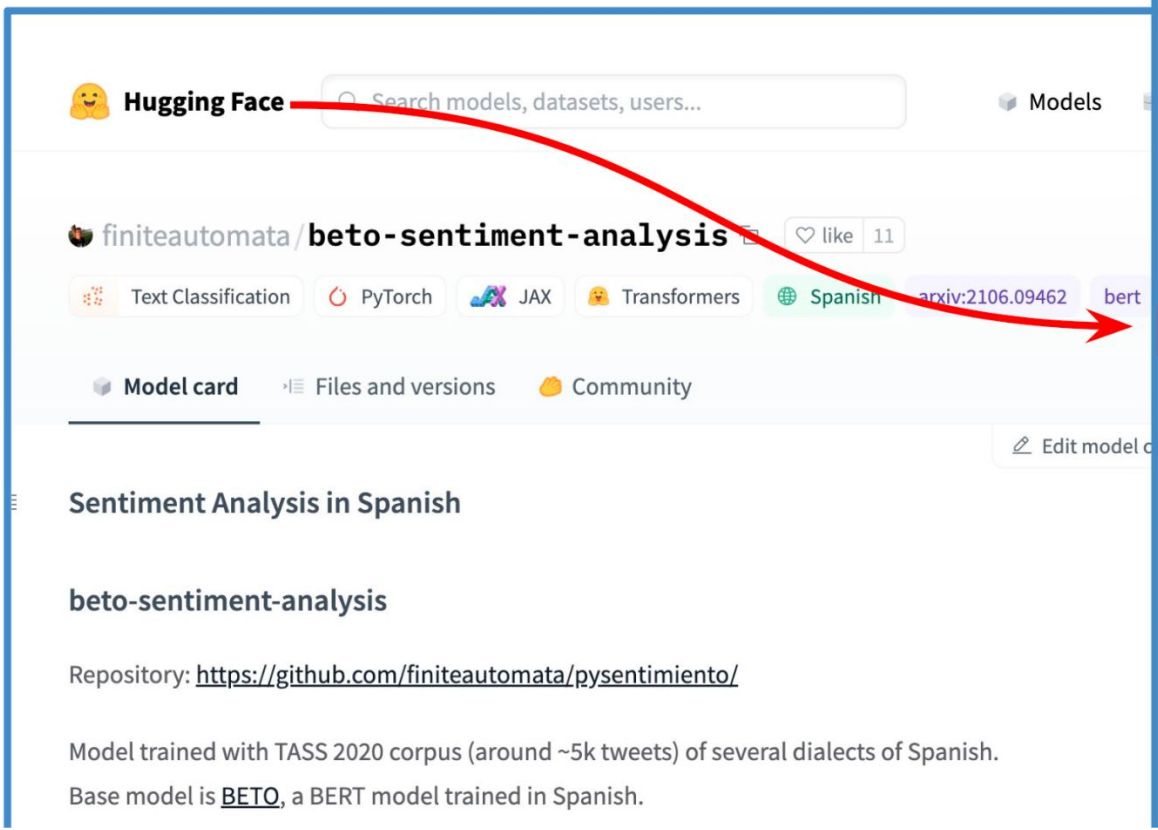
Spark NLP
for Data Scientists

John Snow LABS

Timeline of Large Language Models



Import a model from HuggingFace 🤗 into Spark NLP 🚀



Example Notebooks

HuggingFace to Spark NLP

Spark NLP	HuggingFace Notebooks	Colab
BertEmbeddings	HuggingFace in Spark NLP - BERT	Open in Colab
BertSentenceEmbeddings	HuggingFace in Spark NLP - BERT Sentence	Open in Colab
DistilBertEmbeddings	HuggingFace in Spark NLP - DistilBERT	Open in Colab
CamemBertEmbeddings	HuggingFace in Spark NLP - CamemBERT	Open in Colab
RoBertaEmbeddings	HuggingFace in Spark NLP - RoBERTa	Open in Colab
DeBertaEmbeddings	HuggingFace in Spark NLP - DeBERTa	Open in Colab
XlmRoBertaEmbeddings	HuggingFace in Spark NLP - XLM-RoBERTa	Open in Colab
AlbertEmbeddings	HuggingFace in Spark NLP - ALBERT	Open in Colab
XlnetEmbeddings	HuggingFace in Spark NLP - XLNet	Open in Colab
LongformerEmbeddings	HuggingFace in Spark NLP - Longformer	Open in Colab
BertForTokenClassification	HuggingFace in Spark NLP - BertForTokenClassification	Open in Colab
DistilBertForTokenClassification	HuggingFace in Spark NLP - DistilBertForTokenClassification	Open in Colab
AlbertForTokenClassification	HuggingFace in Spark NLP - AlbertForTokenClassification	Open in Colab
RoBertaForTokenClassification	HuggingFace in Spark NLP - RoBertaForTokenClassification	Open in Colab
BertForSequenceClassification	HuggingFace in Spark NLP - BertForSequenceClassification	Open in Colab
AlbertForSequenceClassification	HuggingFace in Spark NLP - AlbertForSequenceClassification	Open in Colab
RoBertaForSequenceClassification	HuggingFace in Spark NLP - RoBertaForSequenceClassification	Open in Colab
XlmRoBertaForSequenceClassification	HuggingFace in Spark NLP - XlmRoBertaForSequenceClassification	Open in Colab
XlnetForSequenceClassification	HuggingFace in Spark NLP - XlnetForSequenceClassification	Open in Colab
LongformerForSequenceClassification	HuggingFace in Spark NLP - LongformerForSequenceClassification	Open in Colab
AlbertForQuestionAnswering	HuggingFace in Spark NLP - AlbertForQuestionAnswering	Open in Colab
BertForQuestionAnswering	HuggingFace in Spark NLP - BertForQuestionAnswering	Open in Colab
DeBertaForQuestionAnswering	HuggingFace in Spark NLP - DeBertaForQuestionAnswering	Open in Colab
DistilBertForQuestionAnswering	HuggingFace in Spark NLP - DistilBertForQuestionAnswering	Open in Colab
LongformerForQuestionAnswering	HuggingFace in Spark NLP - LongformerForQuestionAnswering	Open in Colab
RoBertaForQuestionAnswering	HuggingFace in Spark NLP - RoBertaForQuestionAnswering	Open in Colab
XlmRobertaForQuestionAnswering	HuggingFace in Spark NLP - XlmRobertaForQuestionAnswering	Open in Colab

TF Hub to Spark NLP

Spark NLP	TF Hub Notebooks	Colab
BertEmbeddings	TF Hub in Spark NLP - BERT	Open in Colab
BertSentenceEmbeddings	TF Hub in Spark NLP - BERT Sentence	Open in Colab
AlbertEmbeddings	TF Hub in Spark NLP - ALBERT	Open in Colab

[Import Transformers into Spark NLP]

Training & Importing Transformers

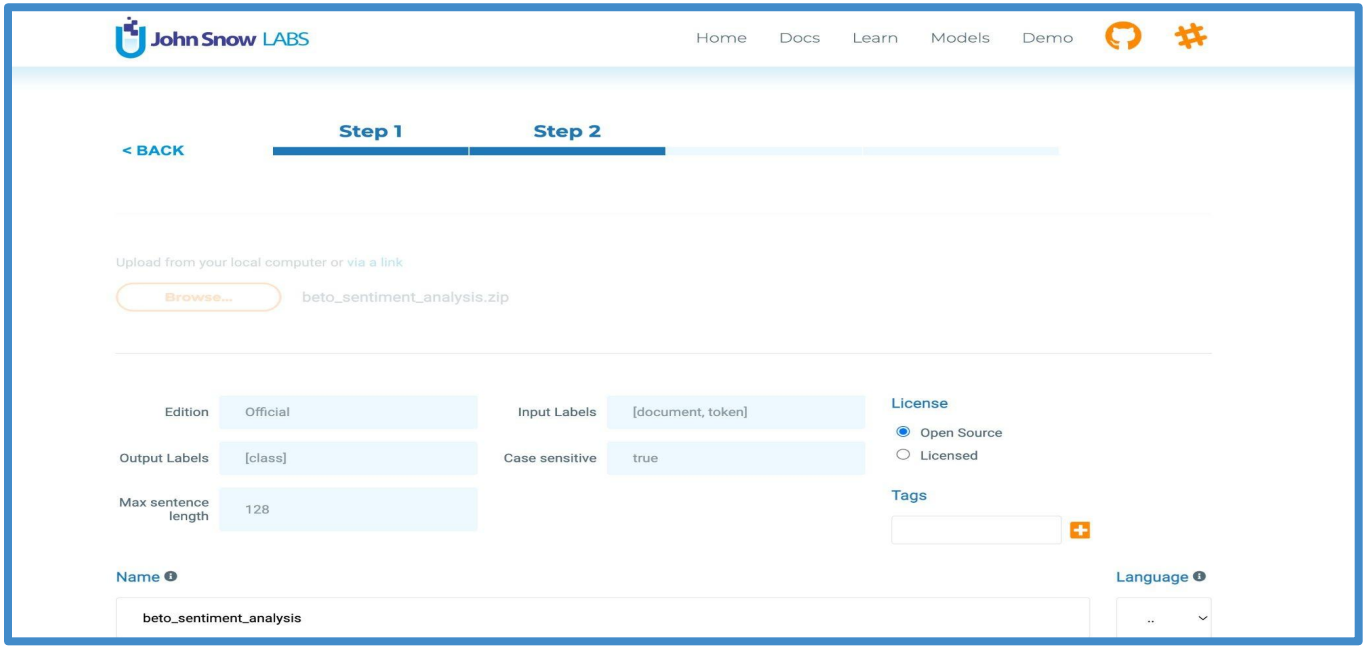
[Spark NLP - Transformers]

Import Transformers into Spark NLP

We have extended support for `HuggingFace` 🤗 and `TF Hub` exported models since `3.1.0` to equivalent Spark NLP 🚀 annotators. Starting this release, you can easily use the `saved_model` feature in HuggingFace within a few lines of codes and import any `BERT`, `DistilBERT`, `CamemBERT`, `RoBERTa`, `DeBERTa`, `XLM-RoBERTa`, `Longformer`, `BertForTokenClassification`, `DistilBertForTokenClassification`, `AlbertForTokenClassification`, `RoBertaForTokenClassification`, `DeBertaForTokenClassification`, `XlmRoBertaForTokenClassification`, `XlnetForTokenClassification`, `LongformerForTokenClassification`, `CamemBertForTokenClassification`, `CamemBertForSequenceClassification`, `CamemBertForQuestionAnswering`, `BertForSequenceClassification`, `DistilBertForSequenceClassification`, `AlbertForSequenceClassification`, `RoBertaForSequenceClassification`, `DeBertaForSequenceClassification`, `XlmRoBertaForSequenceClassification`, `XlnetForSequenceClassification`, `LongformerForSequenceClassification`, `AlbertForQuestionAnswering`, `BertForQuestionAnswering`, `DeBertaForQuestionAnswering`, `DistilBertForQuestionAnswering`, `LongformerForQuestionAnswering`, `RoBertaForQuestionAnswering`, `XlmRoBertaForQuestionAnswering`, `TapasForQuestionAnswering`, `Vision Transformers (ViT)`, `HubertForCTC`, `SwinForImageClassification`, and `ConvNextForImageClassification` models to Spark NLP. We will work on the remaining annotators and extend this support to the rest with each release 😊

Upload trained models to Models Hub

- Trained models can be uploaded and shared via the modelshub
- Zip and Download the model
- Go to https://modelshub.johnsnowlabs.com/ and upload a zip file



Session 4 (Day 2) - Coding Time

- OpenAI in Spark NLP
- Llama2 in Spark NLP
- Import Transformers into SparkNLP

Spark NLP
for Data Scientists

John Snow LABS

Thank You!

Spark NLP Resources

Spark NLP Official page
Spark NLP Workshop Repo
JSL Youtube channel
JSL Blogs
Introduction to Spark NLP: Foundations and Basic Components (Part-I)
Introduction to: Spark NLP: Installation and Getting Started (Part-II)
Named Entity Recognition with Bert in Spark NLP
Text Classification in Spark NLP with Bert and Universal Sentence Encoders
Spark NLP 101 : Document Assembler
Spark NLP 101: LightPipeline
https://www.oreilly.com/radar/one-simple-chart-who-is-interested-in-spark-nlp/
https://blog.dominodatalab.com/comparing-the-functionality-of-open-source-natural-language-processing-libraries/
https://databricks.com/blog/2017/10/19/introducing-natural-language-processing-library-apache-spark.html
https://databricks.com/fr/session/apache-spark-nlp-extending-spark-ml-to-deliver-fast-scalable-unified-natural-language-processing
https://medium.com/@saif1988/spark-nlp-walkthrough-powered-by-tensorflow-9965538663fd
https://www.kdnuggets.com/2019/06/spark-nlp-getting-started-with-worlds-most-widely-used-nlp-library-enterprise.html
https://www.forbes.com/sites/forbestechcouncil/2019/09/17/winning-in-health-care-ai-with-small-data/#1b2fc2555664
https://medium.com/hackernoon/mueller-report-for-nerds-spark-meets-nlp-with-tensorflow-and-bert-part-1-32490a8f8f12
https://www.analyticsindiamag.com/5-reasons-why-spark-nlp-is-the-most-widely-used-library-in-enterprises/
https://www.oreilly.com/ideas/comparing-production-grade-nlp-libraries-training-spark-nlp-and-spacy-pipelines
https://www.oreilly.com/ideas/comparing-production-grade-nlp-libraries-accuracy-performance-and-scalability
https://www.infoworld.com/article/3031690/analytics/why-you-should-use-spark-for-machine-learning.html
https://www.udemy.com/course/spark-nlp-for-data-scientists/

John Snow LABS