Spark NLP: Natural Language Understanding at Scale

Patrick Xavier Marquez Choque*
Universidad Católica San Pablo

Jean Carlo Cornejo Cornejo†
Universidad Católica San Pablo

1 Introduction

El procesamiento de lenguaje natural (NLP) es un componente clave de muchos sistemas de Ciencia de Datos y Big Data, tiene un gran número de aplicaciones para Inteligencia Artificial así que gracias a la popularidad que trae NLP dentro del área de Ciencia de Datos surge la necesidad por conformar una estructura robusta, que pueda escalar fácilmente dentro de un sistema distribuido y este disponible para un gran público de manera *open-source*, es así como surge la creación de Spark NLP desarrollado por David Talby [Kocaman and Talby 2021]

Spark NLP esencialmente es una librería dedicada al procesamiento del lenguaje natural construida sobre Apache Spark y provee operaciones de NLP simples, precisos y con alto rendimiento sobre sistemas distribuidos con más de 1100 pipelines pre-entrenados y modelos desarrollados en más de 192 lenguajes. También es altamente utilizada por organizaciones para el cuidado de la salud teniendo un aumento notable de la popularidad y el uso desde el año 2020.

2 Librería Spark NLP

La librería NLP tiene la capacidad de extraer información y clasificarla dentro de texto en una base de datos que incluso algunos miembros de la comunidad han apoyado con código abierto y que son relativamente populares así qeu las principales herramientas disponibles dentro de Spark NLP son:

- NLTK: Kit de herramientas para el lenguaje natural completo de Spark NLP.
- TextBlob es una API de herramientas construida sobre NLPK y Pattern
- SpaCy es una herramienta orientada para el uso industrial de python y cython.
- Núcleos principales de herramientas NLP de Standford para servicios mucho más orientados a paquetes extensos.
- Fasttext es una sub-librería para el aprendizaje de *word embeddings* y clasificaciones de *sentences* creado por el equipo de investigación de Facebook AI (FAIR)
- Obviamente, hay muchas más bibliotecas en el campo general de NLP y unas cuantas otras sub-librerías dedicadas a casos muchos más específicos como es el caso de la inverstigación biológica para el caso de infecciones que tiene esta librería.

3 Impacto de Spark NLP

Existen campos de investigación los cuales se han visto estancados a nivel de desarrollo de soluciones de software, como el campo orientado al reconocimiento de documentos de salud, es por ello que Spark NLP se plantea como solución debido a que en el área no se utilizaban las tecnologías recientes, incluyendo dentro de estas Machine Learning. Por ello se realizaron diferentes maneras de llegar a una solución a este problema en concreto a través de diferentes modelos entrenados en la clasificación de datos importantes y llegar a una mayor coherencia. Además se determinó que el desempeño del modelo ha probado ser mejor a nivel de cluster comparado que solo una pc teniendo una mejora de 20x a nivel de tokenización y 3.5x a nivel de extracción de entidad.

4 Colaboración de Spark NLP

Como medio principal, la organización ha entregado licencias a alumnos graduados y de pre-grado para que puedan realizar proyectos y experimenten con la tecnología en cuestión y permitir este tipo de contribución para la comunidad de investigadores y facilitar los medios para obtener buenos resultados, siendo el mayor enfoque las empresas orientadas a la salud.

Name	Spark NLP	spaCy	NLTK	CoreNLP
Sentence detection	Yes	Yes	Yes	Yes
Tokenization	Yes	Yes	Yes	Yes
Stemming	Yes	Yes	Yes	Yes
Lemmatization	Yes	Yes	Yes	Yes
POS tagger	Yes	Yes	Yes	Yes
NER	Yes	Yes	Yes	Yes
Dependency parse	Yes	Yes	Yes	Yes
Text matcher	Yes	Yes	No	Yes
Date matcher	Yes	No	No	Yes
Chunking	Yes	Yes	Yes	Yes
Spell checker	Yes	No	No	No
Sentiment detector	Yes	No	No	Yes
Pretrained models	Yes	Yes	Yes	Yes
Training models	Yes	Yes	Yes	Yes

Figure 1: *Servicios/Usos que provee cada librería orientada a NLP.*

En este caso podemos observar las diferentes funciones que ofrecen las librerías relacionadas a NLP disponibles para los usuarios y como en ciertos casos, algunas de estas funciones no están disponibles a excepción de Spark.

5 Metodología de Spark NLP

La librería de Spark NLP trabaja como una librería dedicada para el procesamiento de texto con actualmente 2 versiones, una de ella es open-source y está disponible en un repositorio en Github y la otra versión es enterprise. Cada una de estas versiones viene con un grupo de modelos pre-entrenados y pipelines que pueden ser utilizados en diversos campos.

*e-mail:patrick.marquez@ucsp.edu.pe
†e-mail:jean.cornejo@ucsp.edu.pe

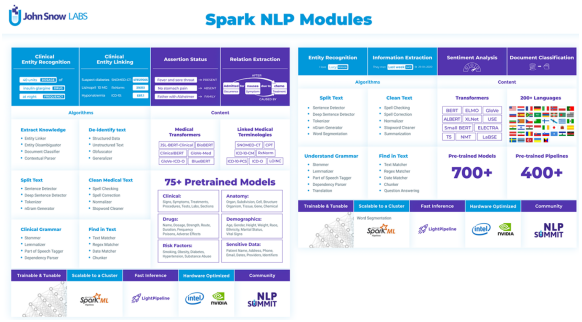

Figure 2: *Módulos del Pipeline de Spark NLP [Kocaman and Talby 2021]*

Como podemos apreciar en la figura 2 podemos ver todos los módulos del Pipeline de la librería Spark NLP. Ya que Spark NLP fue creado a partir de Apache Spark tiene un ecosistema similar al de Apache Hadoop contando con un potente motor de código abierto que proporciona procesamiento de flujo en tiempo real, procesamiento interactivo, procesamiento de gráficos, procesamiento en memoria, así como procesamiento por batches con gran velocidad y facilidad.

Spark tiene un módulo llamado Spark ML que presenta varios componentes de ML que son:

- Estimadores: Son algoritmos entrenables, y transformadores que son el resultado del entrenamiento de un estimador o un algoritmo que no requiere entrenamiento. Tanto los estimadores como los transformadores pueden ser parte de un Pipeline para la utilización de Spark NLP.
- Anotadores de NLP que se fusionan dentro de este marco y sus algoritmos están destinados a predecir en paralelo. Este estimador se ajusta a un DataFrame para producir un transformador de la data entrenandose y así producir el modelo.
- También los Anotadores de Spark NLP son de 2 tipos: AnnotatorApproach y AnnotatorModel. AnnotatorApproach está destinado a ser entrenados a través de la función fit(), y AnnotatorModel extiende los transformadores utilizando la función transform().

Adicionalmente Spark NLP también cuenta con Modelos NLP pre-entrenados listos para su utilización sin la necesidad de realizar ajustar y con la utilización de datos de entrada para realizar sus operaciones. Spark NLP ofrece los siguientes modelos previamente entrenados en cuatro idiomas (inglés, francés, alemán, italiano) y todo lo que necesita hacer es cargar el modelo previamente entrenado en su disco especificando el nombre del modelo y luego configurando los parámetros del modelo como según su caso de uso y conjunto de datos. Entonces no tendrá que preocuparse por entrenar un nuevo modelo desde cero y podrá disfrutar de los algoritmos SOTA pre-entrenados directamente aplicados a sus propios datos con la función transform() .

6 Resultados

Se realizaron 3 experimentos utilizando la librería Spark NLP, el primero es una configuración simple de la instalación del Pipeline de Spark utilizando los modelos pre-entrenados para obtener la clasificación, reconocimiento de texto y análisis de sentimiento; luego se realizó una configuración mucho más detallada de un algoritmo para la categorización de un gran texto, en este caso se utilizó la misma información textual del paper base de Kocaman et al. [Kocaman and Talby 2021] para realizar el análisis de texto en base al pipeline de Spark siguiendo los lineamientos para construir el modelo del analizador sintáctico y por último unos cuantos ejemplos de reconocimiento del idioma de un texto y de análisis de sentimiento utilizando los modelos pre-entrenados para realizarlos de una manera mucha más sencilla.

6.1 Primer Experimento

El primer experimento consta de la instalación de la propia librería y la utilización del modelo utilizando los comandos de la siguiente manera:

Listing 1: *Primer Experimentos Spark NLP*

```
!wget http://setup.johnsnowlabs.com/colab.sh -O ↩
    - | bash
## Con este comando podremos instalar la ↩
    libreria Spark NLP
import sparknlp
spark = sparknlp.start()
## Inicializaremos nuestro Pipeline de Spark
print("Spark NLP version: {}".format(sparknlp.↩
    version()))
print("Apache Spark version: {}".format(spark.↩
    version))
## Verificaremos que estemos utilizando la ↩
    version Spark NLP 3.3.4
from sparknlp.pretrained import ↩
    PretrainedPipeline
## Cargaremos nuestro modelo pre-entrenado para ↩
    reconocimiento
pipeline = PretrainedPipeline('↩
    recognize_entities_dl', 'en')
## Asignaremos este modelo a nuestro pipeline
result = pipeline.annotate('President Biden ↩
    represented Delaware for 36 years in the U.S↩
    .')
## Cargaremos nuestra funcion para almacenar la ↩
    siguiente frase.
print(result['ner'])
print(result['entities'])
## Podemos verificar los resultados reconoce ↩
    tanto a:
## ['O', 'B-PER', 'O', 'B-LOC', 'O', 'O', 'O', '↩
    O', 'O', 'B-LOC']
## 'Biden', 'Delaware', 'U.S', como entidades
pipeline = PretrainedPipeline('↩
    onto_recognize_entities_bert_tiny', 'en')
## Ahora cargaremos nuestro modelo pre-entrenado↩
     para clasificacion
result = pipeline.annotate("Johnson first ↩
    entered politics when elected in 2001.")
## Asignaremos este modelo a nuestro pipeline
print(result['ner'])
print(result['entities'])
## Podemos verificar los resultados reconoce ↩
    tanto a:
## ['B-PERSON', 'B-ORDINAL', 'O', 'O', 'O', 'O',↩
     'O', 'B-DATE']
## 'Johnson', 'first', '2001' como entidades
pipeline = PretrainedPipeline('↩
    analyze_sentimentdl_glove_imdb', 'en')
## Ahora cargaremos nuestro modelo pre-entrenado↩
     para el analisis de sentimientos
result = pipeline.annotate("Harry Potter is a ↩
    great movie.")
## Asignaremos este modelo a nuestro pipeline
print(result['sentiment'])
## Verificamos el resultado como positivo
```

6.2 Segundo Experimento

En este experimento se usó la librería Spark NLP, más especifico los diferentes módulos y sus respectivos modelos pre-entrenados con un conjunto de palabras y diferentes secciones, simulado así un *workflow* que podría ser ejecutado sin problema, en el cual se consideró diferentes tamaños de texto en cuanto a contenido pero no se realizaron pruebas de los límites como tal. Esta prueba contempla idioma inglés como español y se obtuvieron diferentes análisis de la librería al texto como:

- Texto tokenizado.
- Texto evaluado a nivel de entidades (NER), con objetivos médicos.
- Separación del texto en oraciones y palabras.
- Calculo de diferentes pesos de las palabras teniendo en cuenta el significado y la importancia en el texto.

Concluida esta prueba se destaca que no es necesario usar alguna configuración extra para el reconocimiento de otro idioma, siempre que la librería lo permita, cómo se menciono esta no es una prueba de estrés y por tanto no se comprobó el tamaño máximo de archivos que pueden ser leídos y procesados, pero se destaca que pueden ser leídos mediante una variable después de la lectura tradicional de python.

6.3 Tercer Experimento

El tercer experimento de manera similar al segundo este se realizó utilizando la librería Spark NLP pero probando los métodos realizados, este se realizó utilizando los modelos pre-entrenados para realizar una mejor detección del idioma de un texto y el análisis de sentimiento.

Todos experimentos se encuentran en este repositorio [1] de Github.

7 Conclusiones y Trabajos Futuros

- Como principal conclusión tenemos que esta es una librería robusta para la realización de NLP de manera sencilla, preparada para workloads pesados separados en clústers e incluso con la fácil recuperación de modelos ya entrenados para hacer mucho más fácil la realización de estos algoritmos de procesamientop de lenguaje natural.
- Como principal desventaja con los resultados obtenidos son que estos resultados se probaron con la técnologia de Google Colab haciendo muy díficil la comunicación con otros nodos y por lo tanto experimentar dentro de un entorno distribuido pero que se planea tomar como un trabajo a futuro.

References

KOCAMAN, V., AND TALBY, D. 2021. Spark nlp: Natural language understanding at scale. *Software Impacts 8*, 100058.

[1] https://github.com/patrick03524/Big-Data/tree/main/Proyecto%20Spark%20NLP%20Library