# Comparación de herramientas de procesamiento de textos en español extraídos de una red social para Python

Lorena Talamé, Alejandra Cardoso, Matías Amor[1]

[1] Universidad Católica de Salta,
Campo Castañares s/n, 4400, Salta

{mltalame, acardoso, matiasamor}@ucasal.edu.ar

**Resumen.** El procesamiento de lenguaje natural es un área central en cualquier aplicación de análisis de datos textuales. Muchas de estas aplicaciones se realizan a partir de textos publicados en Internet y muy frecuentemente, a partir de textos de redes sociales. Existen diversas herramientas informáticas de código libre que facilitan el procesamiento de lenguaje natural de diversas lenguas, pero especialmente para el idioma inglés. En este artículo se seleccionaron cinco módulos o librerías para el lenguaje Python, con el objetivo de comparar algunas funciones básicas del procesamiento de textos, aplicadas a textos cortos en español extraídos de una red social. La tokenización y segmentación de oraciones fueron perfectamente realizadas por dichas herramientas. Se encontraron diferencias en etiquetado gramatical pero los resultados se consideran aceptables.

**Keywords:** procesamiento de lenguaje natural, etiquetado gramatical, Python

## 1. Introducción

Con el uso de internet y redes sociales, compartimos información (multimedial o textual) que puede resultar valiosa en diversos ámbitos. Muchas personas utilizan las redes sociales para compartir su opinión referida a un evento, a un producto o a una situación en particular. Por lo tanto, resulta interesante poder obtenerlas y analizarlas automáticamente.

Por otro lado, la proporción de la población hispanohablante va en aumento. El español, en el mundo y como lengua materna, ocupa el segundo lugar, luego del chino mandarín [1]. Si nos centramos en los usuarios de Internet, las lenguas que más se emplean son el inglés, el chino y, en el tercer puesto, el español. Sin embargo, si se contempla que el chino, en general, lo usan los nativos, el español se situaría en segundo lugar tras el inglés.

El procesamiento de lenguaje natural (abreviado PLN o en sus siglas en inglés NLP, natural language processing) estudia la interacción entre una computadora y una persona facilitando la comunicación en lenguas naturales o humanas, como el idioma inglés o el español. En la actualidad, es un área de investigación y producción científica muy activa con numerosas aplicaciones.

Existen diversas herramientas informáticas que facilitan el procesamiento de texto en múltiples idiomas, especialmente para el idioma inglés. En este artículo se analizan algunas herramientas de código libre para el lenguaje Python, poniendo el acento especialmente en el análisis morfológico (determinar la forma o tipo gramatical de cada palabra o token en una oración) de textos cortos en lenguaje español, extraídos de la red social Twitter. El trabajo descrito en este artículo forma parte de un proyecto de mayor alcance que aún está en desarrollo.

## 2. Procesamiento de lenguaje

El procesamiento de lenguaje natural es un área de la inteligencia artificial que estudia la comunicación entre las personas y las computadoras utilizando lenguas naturales. Los lenguajes naturales son aquellos que utilizan los humanos para comunicarse ya sea en forma escrita u oral. Algunas aplicaciones típicas del PLN son la búsqueda de respuestas, corrección ortográfica, reconocimiento de voz, generación automática de resúmenes, traducción automática y análisis de sentimientos, entre otras. El proceso del PLN, generalmente, contempla cuatro tipos de análisis sobre la entrada de un texto en lenguaje natural (Fig. 1).

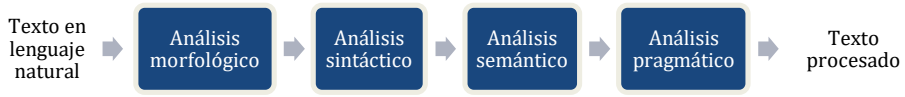


Fig. 1. Proceso PLN

A continuación, se describen los análisis indicados en la figura anterior:

- *análisis morfológico*: consiste en el análisis de las oraciones para identificar unidades léxicas o tokens (unidades del lenguaje con significación propia como palabras o signos de puntuación) y tipos de palabras (sustantivos, verbos, etc.). Una misma palabra puede adoptar diferentes roles en función del contexto en el que aparece, ocasionando problemas de ambigüedad. En el análisis morfológico se determina la forma, clase o categoría gramatical de cada palabra, es decir, se asigna a cada palabra de la oración, el rol que cumple dentro de ella. Esto se conoce como etiquetado gramatical o *POS* (Part-of-speech) *tagging*, el cual debe resolver las ambigüedades que presentan algunas palabras según el contexto donde se encuentren.

- *análisis sintáctico*: el objetivo de este análisis es determinar las relaciones de dependencia estructural entre las palabras de la oración estudiada. Esta dependencia y el orden de las palabras contribuyen a su significado. En la mayoría de las lenguas, este significado se transmite por la sintaxis. Por ejemplo, las dos oraciones: "*A ella le gusta Miguel de Cervantes*" y "*A Miguel de Cervantes le gusta ella*" difieren sólo en términos de sintaxis, pero transmiten significados diferentes. En este nivel se realiza

la desambiguación sintáctica de palabras que pueden funcionar como múltiples partes del discurso. La automatización de esta tarea se dificulta por la ambigüedad del lenguaje [2].

- *análisis semántico*: incluye la desambiguación de palabras con múltiples sentidos. Si se requiriera información del resto de la oración para esta desambiguación, se realizaría en este nivel. Se puede implementar una amplia gama de métodos para lograr la desambiguación, algunos de los cuales requieren información sobre la frecuencia con que ocurre cada sentido de las palabras, en un corpus particular de interés, y otros utilizan el conocimiento pragmático del dominio del documento en cuestión.

- *análisis pragmático*: es uno de los más complejos junto con el semántico. En muchas ocasiones las oraciones, y las palabras que las componen, no pueden ser interpretadas de manera literal ya que su significado depende en gran medida del contexto en el que se utilicen. Entonces, el análisis pragmático, a diferencia del sintáctico y semántico, va un paso más allá de la estructura del lenguaje, ya que es necesario entender el contexto donde las oraciones son emitidas para otorgarles significado.

Cabe mencionar que no siempre se aplican todos los niveles de análisis a cualquier tarea de PLN, depende del objetivo de la aplicación. En este trabajo nos enfocamos en el análisis morfológico.

Generalmente, el PLN necesita algunas acciones de "normalización" de los textos para poder "estructurarlos" y así analizarlos de una manera más fácil. A continuación, se definen algunos de estos procesos.

**Tokenización**

La tokenización es el proceso de dividir un documento de texto en sus distintos componentes, descartando los espacios en blancos y saltos de línea. Un token es una cadena de caracteres (palabra o un signo de puntuación) con algún significado en el contexto de un texto. Por ejemplo, el texto "El niño corre" tiene tres tokens: {'El', 'niño', 'corre'}.

Aunque la tokenización puede resultar trivial, no lo es si se considera que existen caracteres que algunas veces pueden ser parte de un token y otras veces pueden resultar ser separadores de tokens. Por ejemplo, el carácter punto puede ser parte de una abreviatura o sigla, pero también puede resultar ser separador de oraciones.

**Segmentación**

Consiste en separar el texto en fragmentos que puedan tratarse de forma independiente. La forma más usual de segmentación es dividir el texto en párrafos u oraciones.

**Etiquetado gramatical**

El etiquetado gramatical o *POS (part-of-speech) Tagging* consiste en la asignación de una etiqueta o categoría gramatical (sustantivo, verbo, adjetivo, etc.) a cada uno de los tokens de una oración según el rol que cumpla dentro de la misma. Por ejemplo,

para la oración "*El niño corre por la plaza*" el resultado del etiquetado POS se observa en la segunda fila de la Tabla 1.

Tabla 1. Ejemplo etiquetado POS

| Token | **El** | **niño** | **corre** | **por** | **la** | **plaza** |
|---|---|---|---|---|---|---|
| Etiquetado POS | Articulo masculino singular | Nombre común masculino singular | Verbo indicativo presente 3ra. persona del singular | Preposición | Articulo femenino singular | Nombre común femenino singular |

Los algoritmos de POS-tagging deben resolver las ambigüedades que presentan algunas palabras según el contexto donde se encuentren. Por ejemplo, en la oración "*El niño ayuda a su hermana*", la palabra "ayuda" se define como verbo; sin embargo, en la oración "*El niño necesita ayuda*" la misma palabra se establece como sustantivo.

Las categorías gramaticales resultan de gran utilidad por la información que proporcionan acerca de una palabra, sus palabras continuas y la forma de interpretarlas. Saber si una palabra es un sustantivo o un verbo dice mucho sobre las palabras junto a ellas y las formas de interpretarlas. Sirve además para encontrar entidades nombradas en textos y se aplica también a otras tareas de extracción de información.

**Reconocimiento de entidades nombradas**

El reconocimiento de entidades con nombres o entidades nombradas (*NER, Named Entity Recognition*) se ocupa de la extracción de unidades de información predefinidas, como nombres de personas, fechas, lugares, organizaciones, etc.

## 3. Herramientas informáticas para PLN

Existe una amplia variedad de herramientas informáticas para el PLN en diversos idiomas. En este trabajo se seleccionaron algunas Open Source que se pueden utilizar en programas Python[1]. La comunidad científica utiliza en mayor medida el lenguaje interpretado Python por lo que sólo nos centraremos en aquellas bibliotecas que se puedan utilizar en este lenguaje de programación. Cabe mencionar que, si bien no todas son nativas para este lenguaje, como Stanford NLP, existen plugins o APIs que permiten enlazarlas. En el contexto de este trabajo se utilizarán los términos "herramientas", "librerías", "bibliotecas" y "paquetes" como sinónimos.

[1] https://www.python.org

### 3.1. NLTK

NLTK[2] (Natural Language Toolkit) es una plataforma para la creación de programas Python para análisis de texto. Como fue creado con fines educativos en el año 2001, permite la realización de proyectos con distintos propósitos y alcance variado. Provee demostraciones paso a paso de distintos algoritmos [3]. Integra más de 50 corpus y un conjunto de librerías para segmentación, tokenización, etiquetado del habla, análisis sintáctico y semántico, entre otros. Para el español muchas de estas opciones no están disponibles.

Su instalación en Python es sencilla debido a que es un paquete que se encuentra en el repositorio del lenguaje. Con una simple instrucción, se instala lo necesario para utilizarlo.

El etiquetador POS de NLTK originalmente fue creado para el idioma inglés. Sin embargo, se lo puede extender al español entrenando el etiquetador con algún corpus que incorpora NLTK en esta lengua o usando un etiquetador externo como el de Stanford. Para este trabajo se seleccionó la primera opción con el corpus cess_esp [4] que tiene 500.000 palabras y 610 archivos.

El etiquetado POS que retorna NLTK se corresponde con el formato EAGLES[3]. Las etiquetas EAGLES codifican todas las características morfológicas existentes para la mayoría de los idiomas europeos, incluido el español. Estas etiquetas consisten en un conjunto de caracteres de longitud variable donde cada uno corresponde a una característica morfológica. El primer carácter en la etiqueta es siempre la categoría POS que, a su vez, determina la longitud y la interpretación de los elementos siguientes [5].

### 3.2. Freeling

Freeling[4] es una librería de código abierto para el análisis de texto (tokenización, análisis morfológico, detección de entidades nombradas, etiquetado POS, etc.) para una variedad de idiomas. Desde su primera versión, tiene soporte para el español. Fue desarrollado por el Centro de Tecnologías y Aplicaciones del Lenguaje y el Habla (TALP) de la Universidad Politécnica de Catalunya. Se puede utilizar y ampliar los recursos lingüísticos por defecto (diccionarios, lexicones, gramáticas, etc.) adaptándolos a dominios específicos.

Su código fue escrito en C++ bajo una arquitectura cliente – servidor. Existen diversas APIs que permiten utilizar Freeling en otros lenguajes, incluido Python. Su instalación no es sencilla, sobre todo en entornos Windows. Sin embargo, subsanado este inconveniente, al importar la API desde el programa Python se puede invocar a todas las funciones implementadas.

El analizador morfológico de Freeling realiza el etiquetado POS usando las etiquetas EAGLES.

---

[2] https://www.nltk.org/
[3] http://blade10.cs.upc.edu/freeling-old/doc/tagsets/tagset-es.html
[4] http://nlp.lsi.upc.edu/freeling/node/1

### 3.3. Pattern.es

Pattern.es[5] es una biblioteca para Python que contiene herramientas para la conjugación de verbos, la singularización o la pluralización de sustantivos, la división de chunks y permite realizar el etiquetado POS para español. En la versión para el idioma inglés adiciona una interfaz para WordNet. Su instalación y uso son muy sencillos.

Para el etiquetado gramatical utiliza Penn TreeBank Tags [6]. Penn TreeBank está basado en el corpus Brown, pionero en etiquetado POS para inglés. A pesar de haberse creado para el inglés, Pattern lo usa también para el español. Cada etiqueta está formada con dos o tres caracteres que indican la función que cumple cada palabra en una oración.

### 3.4. SPACY

SpaCy[6] es una biblioteca para el procesamiento avanzado de lenguaje natural en Python. Incluye modelos estadísticos pre-entrenados y vectores de palabras, admite tokenización para más de 45 idiomas. Provee etiquetado, análisis y reconocimiento de entidades nombradas y una fácil integración de aprendizaje profundo. Es muy sencilla de utilizar, basta con incorporar desde un programa Python el modelo para el español.

Devuelve un etiquetado POS completo, no sólo indica la función de la palabra en la oración, sino otros datos tales como tiempo verbal, persona, número, modo, género, entre otros. La tokenizacion y el etiquetado gramatical se basan en el corpus OntoNotes5[7] que sigue la sintaxis de Penn TreeBank.

### 3.5. Stanford NLP

Stanford NLP es un framework diseñado y desarrollado para Java que provee soluciones para la mayoría de las tareas más comunes en el procesamiento de lenguaje natural que van desde la tokenización hasta el reconocimiento del discurso [7]. No todas las funciones están implementadas para el español.

Existen APIs para otros lenguajes de programación, entre ellos, Python[8]. NLTK incluye algunas de sus funciones, aunque si se desea hacer un análisis más profundo, siempre se debe correr el servidor de Java en segundo plano y prescindir de NLTK.

Permite realizar etiquetado POS con el formato EAGLES y reconoce entidades para el análisis morfológico. Para el análisis sintáctico, identifica grupos lingüísticos o chunks y se puede obtener el árbol de dependencia.

Su instalación no es tan sencilla como otras de las herramientas analizadas por el hecho de tener que instalar el servidor Java y definir variables de entorno.

---

[5] https://www.clips.uantwerpen.be/pages/pattern-es
[6] https://spacy.io/
[7] https://catalog.ldc.upenn.edu/LDC2013T19
[8] https://stanfordnlp.github.io/stanfordnlp/

### 3.6. Comparación

De cada herramienta se tuvieron en cuenta algunos aspectos como la facilidad en su instalación, sobre todo para una persona con conocimientos básicos o nulos en programación; la necesidad de utilizar una interfaz de comunicación; la presencia de funciones específicas para segmentación y tokenización; el tipo de etiquetado POS que realizan y la implementación de lematizador para el lenguaje español.

A modo de resumen se presenta un cuadro comparativo de algunas funciones básicas de procesamiento de texto y las herramientas mencionadas que fueron objeto de evaluación (Tabla 2).

Tabla 2. Comparación de herramientas

| Herramienta | Código | Segmentación y tokenización | Etiquetado POS | Lematización |
|---|---|---|---|---|
| NLTK | Nativo Python | Si | Eagles | No |
| Freeling | API | Si | Eagles | Si |
| Pattern.es | Nativo Python | Si | Penn TreeBank | Si |
| Spacy | Nativo Python | Si | Penn TreeBank | Si |
| Stanford NLP | API | Si | Eagles | Si |

En la segunda columna de la Tabla 2 se muestra la relación de las herramientas con el lenguaje Python. NLTK, Pattern.es y Spacy se pueden instalar muy fácilmente de los repositorios de Python a través de la línea de comando. No ocurre lo mismo con Stanford NLP y Freeling. Ambos paquetes no son nativos de Python; fueron escritos en Java y C++, respectivamente. La instalación de Stanford es más sencilla que la de Freeling aunque ambos necesitan sendos servidores que se estén ejecutando para funcionar. Sin embargo, si no se requiere de todo el potencial de ambas herramientas, existe otra opción para su uso: Stanford se puede utilizar a través de NLTK y Freeling provee un ejecutable que realiza la segmentación, la tokenización, el etiquetado POS y la lematización recibiendo un archivo como entrada.

Todas las herramientas poseen funciones para segmentación y tokenización. NLTK, Freeling y Stanford NLP realizan el etiquetado POS con etiquetas Eagles mientras que Pattern.es y Spacy lo hacen con etiquetas de Penn TreeBank.

Salvo NLTK, las demás herramientas, implementan un lematizador para el idioma español. El lematizador disponible en NLTK, WordNet, está disponible para el idioma inglés.

## 4. Corpus

Para poder evaluar las diferentes alternativas mencionadas para el procesamiento del lenguaje natural, se diseñó un corpus de textos cortos obtenidos de la red social Twitter (llamados *tweets*), por lo cual los textos que serán objeto de estudio son subjetivos o coloquiales. Se capturaron tweets utilizando la API[9] de Twitter y a partir de una serie de hashtags sobre temas de actualidad de Argentina durante Octubre del año 2018. Los mensajes se almacenaron en una base de datos NoSQL. Se optó por trabajar con mensajes en lenguaje español, generados en nuestro país (según identificador de geolocalización) y se evitó mensajes con errores ortográficos. Si bien se recopilaron muchos tweets durante este mes, se descartaron una gran parte de ellos por contener solo imágenes, íconos o textos con poca información para el análisis posterior. Finalmente se seleccionaron al azar 100 mensajes, algunos de los ellos compuestos por más de una oración.

Como el objetivo de esta experimentación fue comparar las devoluciones de cada herramienta de PLN, en primer lugar, fue necesario realizar una serie de acciones para eliminar íconos, emojis, hashtags y links, los cuales son muy usuales en estos tipos de textos.

### 4.1. Preparación de documentos

Muchos comentarios en las redes sociales, no suelen tener en cuenta las reglas ortográficas, utilizan palabras del lunfardo o abreviaturas y muchas veces acentúan sus emociones con el uso de *emojis* (pequeñas imágenes que representan emociones, comidas, lugares, etc.), palabras completas en mayúsculas o signos de exclamación. Por esta razón, se realizaron una serie de acciones a modo de limpieza de los textos (Fig. 2) previo al análisis, ya que las herramientas analizadas sólo reconocen palabras válidas del idioma español.


Fig. 2. Preparación de documentos

Como primer paso, teniendo en cuenta que el objetivo del trabajo es comparar las herramientas a nivel textual, se eliminaron los emojis, url's y hashtags presentes en los tweets. Para esto se escribió un programa Python y se utilizaron expresiones regulares y la librería *emoji* que detecta símbolos Unicode.

La repetición de letras es usual para intensificar la emoción en un texto. Por ejemplo, "*Es taaaaaan pero tan difícil que yo me aleje de alguien que quería, pero si lo hice es*

[9] https://developer.twitter.com/

*porque de verdad ya no aguantaba más*.". En este caso, la palabra "*taaaaaan*" demuestra el acento que quiso poner el usuario, pero no es una palabra válida en español. Por esto, se descartaron las letras repetidas e innecesarias para tener una palabra reconocida en la lengua española [8]. En el ejemplo, se reemplazaron las seis letras "*a*" por una sola obteniendo como resultado "*tan*" que sí es correcta en esta lengua.

La última tarea realizada fue reemplazar aquellas abreviaciones que comúnmente se utilizan por la palabra completa. Por ejemplo, se reemplazó el "*x*" por "*por*" y "*xq*" por "*porque*", entre otras.

## 5. Experimentos

Para el estudio comparativo se evaluaron las funciones de segmentación de oraciones, tokenización y etiquetado POS en cada herramienta. Con el objeto de medir la capacidad de cada una y de tener un punto de comparación, se aplicaron, manualmente, éstas formas de procesamiento al corpus. Para facilitar las comparaciones posteriores se equipararon las etiquetas EAGLES y Penn TreeBank, definiendo nuevas etiquetas (segunda columna de Tabla 3) con las cuales se etiquetó el corpus. Las columnas siguientes de la tabla, contienen las etiquetas utilizadas por los analizadores (EAGLES y Penn TreeBank) y ejemplos. Nótese que de EAGLES solo se utilizaron los primeros caracteres que son los que contienen la información principal de cada etiqueta.

Tabla 3. Equivalencia de etiquetas

| Descripción | Etiqueta | EAGLES | TreeBank | Ejemplo |
|---|---|---|---|---|
| Conjunción | CONJ | CC | CC | y, o |
| Determinante | DET | DA | DT | la, el |
| Adjetivo | ADJ | AQ | JJ | alegre, mala |
| Verbo | VERBO | VM, VA, VS | VB, VBZ, VBP | estoy, suman |
| Sustantivo | SUST | NC | NN, NNS | circo, noche |
| Nombre Propio | NPRO | NP | NP, NPS | Argentina |
| Adverbio | ADV | RG | ADP | siempre, pronto |
| Interjección | INTJ | I | IN | ah, ay |
| Signos Puntuación | PUNT | FAT, FP, FC | O | . ; ! |

### 5.1. Segmentación

El corpus está compuesto de 136 oraciones (algunos tweets contienen más de una oración) muchas de las cuales no estaban correctamente separadas dentro de cada tweet, es decir, luego de punto al final de oraciones no se encontró un espacio antes del comienzo de la oración siguiente, por ejemplo: "*Nosotros vivíamos en el campo.Nos relacionábamos con familias amigas*". Otro problema encontrado en el corpus es que muchos tweets después de un punto no comenzaban con mayúsculas. Un ejemplo de esto es "*Mi papá se jubila e invita a los vecinos a un costillar. tremenda felicidad*". Además, a otros comentarios les faltaba el punto para diferenciar una sentencia de otra. Para algunos textos, NLTK detectó más oraciones de las que realmente tienen. Estos son los casos que contienen más de un signo de puntuación consecutivo. Por ejemplo: *"¡¡¡Vacaciones!!! Lo bueno es que desperté y ya sabía que era domingo.*" Detectó tres oraciones: *"¡¡¡Vacaciones!!*", *"!*" y *"Lo bueno es que desperté y ya sabía que era domingo.* "

En primer lugar, se evaluó la capacidad de las herramientas para la segmentación. A pesar de los inconvenientes mencionados, todas las herramientas lograron distinguir las oraciones en un porcentaje aceptable, Stanford NLP logró el mejor resultado segmentando el 98,53% de las oraciones. En la segunda columna de la Tabla 4 se observa, la cantidad de oraciones detectadas por cada analizador y en la última columna los porcentajes correspondientes.

Tabla 4. Resultados segmentación

| **Oraciones en corpus** | **136** | |
|---|---|---|
| Stanford NLP | 134 | 98.53% |
| NLTK | 129 | 94.85% |
| SPACY | 124 | 91.18% |
| Freeling | 123 | 90.44% |
| Pattern.es | 128 | 94.17% |

Para todo el análisis que continúa se solucionaron los problemas de separación de oraciones. De esta manera, dos herramientas lograron detectar la totalidad de sentencias: Stanford NLP y NLTK. Pattern sólo no encontró una. Freeling aumentó su rendimiento detectando el 95,58% de las oraciones. Spacy sólo encontró cuatro sentencias más que antes logrando un 94,12% de eficiencia. En la Tabla 5 se reflejan estos resultados.

### 5.2. Tokenización

Nltk, Spacy y Pattern no tokenizaron correctamente aquellas palabras precedidas por un signo de exclamación o interrogación. Por ejemplo, en la oración *"¡No creo en el circo de la información!*", se consideró "*¡No*" como un único token cuando lo correcto son dos tokens: "*¡*" y "*No*".

En el corpus se identificaron 1481 tokens (palabras, signos de puntuación, números y fechas), de los cuales más del 90% fueron correctamente identificados por todas las herramientas.

Tabla 5. Resultados de segmentación con textos corregidos

| **Oraciones en corpus** | **136** | |
|---|---|---|
| Stanford NLP | 136 | 100% |
| NLTK | 136 | 100% |
| SPACY | 128 | 94.12% |
| Freeling | 130 | 95.58% |
| Pattern.es | 135 | 99.26% |

### 5.3. Etiquetado POS

Las herramientas utilizan distintas nomenclaturas para el etiquetado POS pero, como se mencionó antes, se realizó una equiparación para permitir la comparación. En la Tabla 6 se observa el resultado de la evaluación de la oración *"¡No creo en el circo de la información!*". En la segunda fila se observa el etiquetado realizado manualmente y en las filas posteriores las etiquetas devueltas por cada analizador (equiparadas según se describe en la Tabla 3). Obsérvese que "DESC" representa la etiqueta "None" que devuelve Nltk al no poder clasificar.

Tabla 6. Ejemplo etiquetado POS

| **Tweet** | ***¡*** | ***No*** | ***creo*** | ***en*** | ***el*** | ***circo*** | ***de*** | ***la*** | ***información*** | ***!*** |
|---|---|---|---|---|---|---|---|---|---|---|
| | **PUNT** | **ADV** | **VERBO** | **PREP** | **DET** | **SUST** | **PREP** | **DET** | **SUST** | **PUNT** |
| **Nltk** | DESC | | VERBO | PREP | DET | SUST | PREP | DET | SUST | PUNT |
| **Freeling** | PUNT | ADV | VERBO | PREP | DET | SUST | PREP | DET | SUST | PUNT |
| **Pattern** | NPRO | | VERBO | INTJ | DET | SUST | INTJ | DET | SUST | PUNT |
| **Spacy** | SUST | | VERBO | ADV | DET | SUST | ADV | DET | SUST | PUNT |
| **Stanford** | PUNT | ADV | VERBO | PREP | DET | SUST | PREP | DET | SUST | PUNT |

A continuación, se describen algunas particularidades detectadas de cada librería.

**NLTK**

Una de las principales dificultades de esta herramienta es que no logra separar palabras precedidas o seguida de signos de puntuación (punto, exclamación o interrogación). Muchos de los casos analizados fueron etiquetados como "None", perdiendo así información muy útil sobre el etiquetado. Algunos ejemplos: "*cambiare!",* "*cariño?*".

**FREELING**

Si bien en este estudio no se realizó el reconocimiento de entidades nombradas es importante mencionar que Freeling las detecta de manera más acertada que el resto de las herramientas. Por ejemplo, en la oración "*La mejor noche con mis amigos en San Salvador de Jujuy*", Freeling indica que [San_Salvador_de_Jujuy] es un nombre propio (Tabla 7).

Tabla 7. Ejemplo 1

| **Tweet** | ***La*** | ***mejor*** | ***noche*** | ***con*** | ***amigos*** | ***en*** | ***San*** | ***Salvador*** | ***de*** | ***Jujuy*** |
|---|---|---|---|---|---|---|---|---|---|---|
| Nltk | DT | ADJ | SUST | PREP | SUST | PREP | DESC | NPROP | PREP | DESC |
| Freeling | DT | ADJ | SUST | PREP | SUST | PREP | NPROP | | | |
| Pattern | DT | ADJ | SUST | PREP | SUST | INTJ | ADJ | NPROP | INTJ | NPROP |
| Spacy | DT | ADJ | SUST | ADV | SUST | ADV | NPROP | NPROP | ADV | NPROP |
| Stanford | DT | ADJ | SUST | PREP | SUST | PREP | NPROP | NPROP | PREP | NPROP |

Solo algunas palabras (27) que comienzan con mayúsculas las clasifica como nombre propio cuando en realidad no lo son, por ejemplo "*Te Amo*". Este es un problema que encontramos en este tipo de textos, donde no siempre se respetan las reglas ortográficas. Freeling también identifica adverbios compuestos por más de una palabra, por ejemplo: "*por favor*" lo clasifica como adverbio (en este caso, adverbio de modo) mientras que las otras herramientas indican que son preposición y sustantivo (ejemplo en Tabla 8).

Tabla 8. Ejemplo 2

| *Tweet* | ***Por*** | ***favor*** | ***,*** | ***no*** | ***confundas*** | ***a*** | ***la*** | ***gente*** |
|---|---|---|---|---|---|---|---|---|
| Nltk | PREP | SUST | PUNT | ADV | DESC | PREP | DT | SUST |
| Freeling | ADV | | PUNT | ADV | VERBO | PREP | DT | SUST |
| Pattern | INTJ | ADV | PUNT | ADV | SUST | INTJ | DT | SUST |
| Spacy | ADV | SUST | PUNT | ADV | ADJ | ADV | DT | SUST |
| Stanford | PREP | SUST | PUNT | ADV | VERBO | PREP | DT | SUST |

**PATTERN**

En el corpus existen cuatro menciones a días de la semana (ejemplo "*Primera vez que no quiero que llegue el viernes*"). Pattern los clasifica como nombres comunes o sustantivos, sin embargo, las otras herramientas les asignan la etiqueta "fecha".

Al igual que NLTK, no detecta los signos de exclamación e interrogación de apertura. A las preposiciones las identifica como interjecciones.

**SPACY**

A muchas de las preposiciones las clasifica como adverbios y las fechas como sustantivo. Sin embargo, si se incluyen otros argumentos al análisis, se observa que las identifica correctamente, es decir devuelve adtype=prep y advtype= time, respectivamente.

**STANFORD**

Tanto Stanford cono Freeling, a las contracciones las dividen en dos palabras. Por ejemplo, la contracción "del" la separa en "de" y en "el" asignándoles las etiquetas preposición y determinante, respectivamente.

De los diez números existentes en el corpus, solo identifica nueve. Las palabras "sábado" y "viernes" se clasifican como sustantivo, a diferencia de Freeling que les asigna la etiqueta "fecha".

## 6. Resultados

Freeling etiquetó correcta y completamente los tweets, en el 28% de los casos y Stanford en el 19%, el resto de las herramientas no superaron el 6%, y esto se debe, a que en muchos textos Spacy, Nltk y Pattern no diferencian los tokens que comienzan con signos de puntuación. Se observa que modificando los textos para una correcta tokenización, los porcentajes de estas últimas herramientas mejoran, pero no superan

los valores obtenidos por Freeling y Stanford. El resto de los textos tuvieron coincidencia en gran parte del etiquetado, así, todas las herramientas coincidieron con el etiquetado manual entre el 64% y 75% (Tabla 9).

Tabla 9. Resultados evaluación

| | **Nltk** | **Freeling** | **Pattern** | **Spacy** | **Santford** |
|---|---|---|---|---|---|
| Total | 3% | 28% | 4% | 6% | 19% |
| Parcial | 72% | 64% | 64% | 75% | 72% |

## 7. Conclusiones

El procesamiento de lenguaje natural es un área central en cualquier aplicación de análisis de datos textuales. Muchas de estas aplicaciones se realizan a partir de textos publicados en Internet y muy frecuentemente, a partir de textos de redes sociales. Por otro lado, la variedad de herramientas Open Source para procesamiento de textos es amplia, lo cual facilita cualquier tipo de investigación en este campo. En este artículo se utilizaron cinco herramientas o librerías para evaluar sus resultados sobre un corpus de textos cortos, en español y con términos coloquiales.

A diferencia de otros trabajos que realizaron comparaciones de herramientas de PLN, para la creación y explotación de corpus [9] se utilizaron textos escolares o fuentes formales (por ejemplo en [10] se construyó un corpus de sentencias), el corpus de textos de este trabajo fue obtenido de una red social. Los comentarios en redes sociales suelen contener imágenes, links, íconos, emojis, y muchas veces contenido textual incorrecto desde el punto de vista ortográfico. Por esta característica fue necesaria una etapa de preparación previa al análisis de las herramientas.

Se compararon tres funciones básicas del PLN: segmentación de oraciones, tokenización y etiquetado gramatical. Respecto a los dos primeros no se observaron mayores diferencias. Considerando el etiquetado completo y correcto de cada oración, Freeling tuvo la mayor cantidad de aciertos.

Desde el punto de vista de la sencillez de instalación, las herramientas nativas de Python permiten utilizar repositorios lo que facilita su uso a personas no expertas en programación. Tanto Stanford como Freeling, necesitan conocimiento más específico para utilizar sendos servidores.

El resultado de esta comparación será utilizado en otro proyecto que está desarrollando este equipo de investigación. Aunque las herramientas analizadas proveen más opciones para el análisis de textos, nuestro objetivo consistió en evaluar sólo aquellas que serán utilizadas en el proyecto principal.

## Referencias

[1] Instituto Cervantes, «El español: una lengua viva. Informe 2018,» 2018. [En línea]. Available: https://cvc.cervantes.es/lengua/espanol_lengua_viva/pdf/espanol_lengua_viva_2018.pdf. [Último acceso: 02 2019].

[2] S. N. Galia Haro y A. Gelbukh, Investigaciones en análisis sintáctico para el español, México: Instituto Politécnico Nacional, Dirección de Publicaciones, 2007.

[3] S. Bird y E. Loper, «NLTK: The Natural Language Toolkit,» *ACL Association for Computacional Linguistics,* 2002.

[4] M. Martí, M. Taulé, L. Márquez y M. Bertran, «Cess-ece: corpus anotados del español y catalán.,» *Arena Romanistica. A New Nordic Journal of Romance Studies,* vol. 1, 2007.

[5] Research Center on Language and Speech Technologies and Applications, «Freeling 4.1,» [En línea]. Available: https://talp-upc.gitbook.io/freeling-4-1-user-manual/tagsets. [Último acceso: 02 2019].

[6] «Penn Treebank P.O.S. Tags,» [En línea]. Available: https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html. [Último acceso: 02 2019].

[7] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard y D. McClosky, «The Stanford Core Natural Language Processing Toolkit,» *Association of Computer Linguistics,* pp. 55-60, 2014.

[8] S. Brody y N. Diakopoulos, «Cooooooooooooooollllllllllll!!!!!!!!!!!!!! Using Word Lengthening to Detect Sentiment in Microblogs,» *Empur. Methods Nat. Lang. Process.,* pp. 562-570, 2011.

[9] J. Vivaldi, «Catálogo de herramientas informátivas relacionadas con la creación, gestión y explotación de corpus textuales,» *Revista Tradumàtica - Traducció i Tecnologies de la Informació i la Comunicació,* nº 07, p. http://www.fti.uab.cat/tradumatica/revista, 2009.

[10] B. Valverde Olmedo y R. Martinez García, «Herramientas de código abierto para el procesamiento del español. Análisis y comparativa,» *Dialnet. Revista hiberoamericana de lingüística,* nº 11, pp. 287-308, 2016.

[11] E. D. Liddy, Natural Language Processing, 2001.

[12] B. Felbo, A. Mislove, A. Sogaard, I. Rahwan y S. Lehmann, Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm, 2017.

[13] M. P. Marcus, M. A. Marcinkiewicz y B. Santorini, «Building a Large Annotated Corpus of English: The Penn Treebank,» *Journal Computational Linguistics,* vol. 19, pp. 313-330, 1993.