**INSTITUTO TECNOLÓGICO DE BUENOS AIRES – ITBA**

**ESCUELA DE (INGENIERÍA Y TECNOLOGÍA – INGENIERÍA Y GESTIÓN - POSTGRADO)**

# PROCESAMIENTO DE LENGUAJE NATURAL APLICADO A LOS DISCURSOS DE JUAN DOMINGO PERÓN ENTRE 1943 Y 1955

**AUTOR/ES: Olmos, Martín (Leg. Nº 104149)**

**DOCENTE/S TITULAR/ES O TUTOR/ES: Gómez, Leticia I.**

**TRABAJO FINAL PRESENTADO PARA LA OBTENCIÓN DEL TÍTULO DE ESPECIALISTA EN CIENCIA DE DATOS**

**BUENOS AIRES**

**PRIMER CUATRIMESTRE, 2021**

*A Catalina, lo más lindo que me pasó en la vida.*

# Table of Contents

# 1. Introducción

El Procesamiento de Lenguaje Natural o NLP (por sus siglas en inglés) es el campo que se dedica al estudio del lenguaje humano mediante el uso de distintas técnicas computacionales y de aprendizaje automático. Estas técnicas permiten, entre otras cosas, encontrar patrones en grandes volúmenes de datos textuales.

Si bien las primeras técnicas de NLP se remontan a los trabajos de Alan Turing en 1950, el gran desarrollo de este campo se dio más recientemente con la aplicación de algoritmos de aprendizaje automático en la década de 1980 y la explosión en la capacidad de procesamiento y almacenamiento resultante de la Ley de Moore a partir de la década de 1990.

El NLP consiste en un conjunto de técnicas de las cuales utilizaremos las siguientes:

- Tokenización
- Lematización
- Extracción de entidades
- *Part-of-speech (POS) tagging* o etiquetado de partes-del-discurso.

El procesamiento de lenguaje natural puede ser aplicado a una amplia gama de problemas. Algunas de las aplicaciones más comunes son las siguientes:

- Recuperación de Información (Information Retrieval): se refiere a sistemas que permiten obtener información específica a partir de una consulta determinada de un usuario en una base de datos con gran cantidad de documentos.
- Extracción de Información (Information Extraction): se refiere al reconocimiento, etiquetado y extracción de cierta información clave para constituir una representación estructurada.
- Respuesta a Preguntas (Question Answering): se trata de la construcción de sistemas que puedan responder preguntas formuladas por humanos en lenguaje natural.
- Resumen Automático (Automatic Summarization): se refiere a la creación de resúmenes de un texto a partir de encontrar un subconjunto de los datos con la información clave del conjunto de datos.

- Traducción Automática (Machine Translation): se refiere a sistemas que traducen de un lenguaje humano a otro sin ningún tipo de asistencia humana.

Por otro lado, el procesamiento de lenguaje natural permite una interacción humano-máquina que habilita aplicaciones como análisis de sentimientos, etiquetado de partes-de-discurso, extracción de tópicos, reconocimiento de entidades nombradas, síntesis automática de textos, extracción de relaciones, stemming, entre otras. Estas técnicas permiten aplicaciones tales como la minería de texto, la respuesta automática de preguntas, la traducción automática, entre otras. Algunas aplicaciones conocidas son:

- Análisis de redes sociales
- Filtrado automático de conversaciones ofensivas
- Seguimiento de tendencias y tópicos importantes
- Chat bots
- Monitoreo de ciber estafas como phishing
- Extracción de etiquetas y palabras clave

En el presente trabajo, utilizaremos técnicas de procesamiento de lenguaje natural a los discursos de uno de los políticos más importantes de la Argentina en el siglo XX: el tres veces electo Presidente de la Nación Juan Domingo Perón.

## 2. Presentación del Problema

La política a menudo se expresa a través de la palabra escrita y hablada. El análisis de todos los discursos políticos suele ser valioso, ya que puede dar cuenta de las tendencias dentro del contexto. Sin embargo, los costos de analizar grandes volúmenes de texto han sido un obstáculo para su uso en la ciencia política.

He aquí la gran utilidad de la computación y su capacidad para reducir enormemente los costos del análisis de textos. La posibilidad actual de contar con técnicas que permiten procesar grandes volúmenes de textos en diversos formatos digitales, hace promisorio el procesamiento que facilite el posterior análisis.

Si bien las técnicas computacionales no están exentas de problemas, y de ninguna manera pueden sustituir una lectura profunda, sí permiten amplificarla y potenciarla

(Grimmer & Stewart, 2013) mediante una gran variedad de métodos que utilizaremos en este trabajo.

# 3. Estado del Arte / Contexto

En esta sección seguiré el texto de Bhargavi Goel, "Developments in The Field of Natural Language Processing" (Goel, 2017).

El Procesamiento de Lenguaje Natural (NLP) es un campo de investigación que busca analizar, entender y manipular el lenguaje de textos escritos y hablados mediante técnicas computacionales.

El NLP como disciplina surgió a fines de la década de 1940. El primer artículo publicado sobre el tema lo publicó Alan Turing en 1950 y se tituló "Computing Machinery and Intelligence" (Turing, 2009), un trabajo que sostenía la importancia que computadoras inteligentes pudieran mantener conversaciones con personas sin que éstas notaran que están hablando con máquinas.

Hasta fines de la década de 1960 la cuestión de la representación del significado y el desarrollo de soluciones manejables computacionalmente no era factible con los marcos teóricos de las teorías de gramática de ese entonces. Esto comienza a cambiar, en relación a las teorías de gramática, con el modelo transformacional de competencia lingüística introducido por Chomsky en 1965, considerado una nave insignia en el desarrollo del campo de NLP. Otros trabajos posteriores vinieron a subsanar los puntos flojos de esta teoría, relacionados con un enfoque muy centrado en lo sintáctico y dificultades para la implementación computacional del modelo.

Hacia fines de la década de 1970 ya se observa una mayor preocupación por temas semánticos y el fenómeno del discurso.

En la década de 1980 el avance en la capacidad de cómputo habilitó aproximaciones basadas en métodos estadísticos. La introducción de técnicas de aprendizaje automático significó una revolución. Comenzó con árboles de decisión para luego pasar a modelos más confiables, integrados a sistemas más amplios. Desarrollos más recientes cambiaron el foco de modelos supervisados a no-supervisados o semi-supervisados.

En la década de 1990 continuó la tendencia hacia el uso de modelos empíricos a partir de textos de lenguaje natural y el advenimiento de corpus más grandes.

El aumento en la capacidad de cómputo y almacenamiento y en la disponibilidad de texto electrónico a través de internet significaron una explosión en el campo de NLP. En los 2000 esta tendencia se consolida y se considera a estos métodos estadísticos muy eficientes y capaces de realizar tareas de análisis de lenguaje con un desempeño similar al de los humanos. Este desempeño similar al de los humanos se logró por ejemplo en los primeros modelos Hidden Markov para la tarea de etiquetado de partes de discurso (Part-Of-Speech tagging). Otro hito importante en esta década se dio con la llegada de los clasificadores bayesianos con aplicaciones en la clasificación de documentos que hizo posible, por ejemplo, analizar correos electrónicos y SMS y etiquetarlos como spam o mensaje no deseado. Además, se introdujeron los modelos n-gram cuya simpleza y escalabilidad llevaron a su gran adopción para el reconocimiento de textos y discursos.

Los enfoques estadísticos fueron exitosos en responder a problemas como la identificación de partes-de-discurso, desambiguación de sentido de palabras, entre otros. La próxima generación de sistemas de NLP que se está desarrollando apuntan a la comprensión de discurso.

Otro quiebre en el campo de NLP ha sido la posibilidad de establecer relaciones semánticas mediante procesamiento no-supervisado de grandes cantidades de texto con el surgimiento de Word2vec.

Otro avance notable se dio con la llegada del macheo aproximado de palabras (fuzzy string matching). Estos algoritmos determinan la distancia entre palabras a través de la cantidad de operaciones primitivas para lograr un macheo exacto. Estas técnicas se utilizan, por ejemplo, en los programas de corrección ortográfica.

Un gran paso adelante en NLP lo representan los dispositivos a los cuales un humano les habla. Una gran cantidad de investigaciones se están desarrollando en el campo de NLP para la mejora de sistemas humanizados, con el fin de hacerlos capaces de entender instrucciones simples.

Finalmente, antes de culminar el repaso del estado del arte, es necesario mencionar un desarrollo reciente que está revolucionando los campos de NLP e inteligencia artificial en general: GPT-3 (Generative Pre-trained Transformers), un modelo de lenguaje autorregresivo que utiliza aprendizaje profundo (Deep Learning) para producir texto

parecido al generado por humanos. Fue desarrollado por OpenAI, una organización sin fin de lucro con la misión de "asegurar que la Inteligencia Artificial General (AGI)... beneficie a la humanidad", de acuerdo a lo señalado en su página web (https://openai.com/about/).

GPT-3 es un modelo de lenguaje pre-entrenado con 175 billones de parámetros, cuya sintonía fina se logra agregando tan sólo unos pocos ejemplos del campo específico de aplicación. GPT-3 logra muy alto rendimiento en muchas tareas de NLP como traducción, respuesta a preguntas, así como tareas que requieren razonamiento sobre la marcha o adaptación a un dominio específico, tales como descifrar palabras con letras mezcladas, agregar una palabra nueva a una oración o hacer operaciones de tres dígitos.

# 4. Objetivos

## 4.1. Objetivo General

El objetivo del presente trabajo consiste en aplicar las técnicas de procesamiento de lenguaje natural a los discursos de uno de los políticos más importantes de la Argentina en el siglo XX: el tres veces electo Presidente de la Nación, Juan Domingo Perón.

## 4.2. Objetivos Específicos

- Recopilar los discursos del ex presidente, para ser utilizados con fuente de datos.
- Aplicar técnicas de ETL para limpiar y pre-procesar el set de datos obtenido.
- Aplicar técnicas de NLP para analizar los discursos y contribuir a la respuesta de las siguientes preguntas: ¿Cómo podemos caracterizar a los discursos del ex presidente en términos de periodicidad, extensión y frecuencia? ¿Cuáles fueron los tópicos preponderantes en los discursos de Perón a lo largo del tiempo? ¿Cuáles fueron las personas, lugares y organizaciones más presentes en sus discursos? ¿Cómo evolucionó la polaridad en los mismos?
- Además de los aportes sustantivos, se pondrá a disposición de la comunidad científica el corpus constituido por los 3.376 discursos que se reunieron,

limpiaron y pre-procesaron como parte del presente trabajo, y que serán publicados en formato abierto junto con el mismo.

# 5. Alcances del Trabajo y Limitaciones

En el presente trabajo se analizarán todos los discursos pronunciados por el ex presidente entre el 01 de diciembre de 1943 y 19 de septiembre de 1955.

Este análisis, que se realizará sobre frecuencias temporales y sobre el contenido, es objetivo sobre los hechos y no incluye análisis sociológico alguno.

# 6. Métodos y Herramientas Utilizados en el Trabajo

## 6.1. Elección de las Fuentes

Los documentos, con excepción de los correspondientes al año 1949, fueron suministrados por el historiador Enrique de Alzáa, quien lideró un equipo que transcribió a formato digital editable los originales en papel que se encuentran en el Archivo General de la Nación. Dado que este trabajo se realizó hace varios años y en distintas épocas, los documentos recibidos corresponden a tres versiones diferentes de documentos de Microsoft Word. Los discursos del año 1949 fueron tomados de Perón (2016) en formato PDF.

En total, contamos con 1163 discursos que van desde 1943-12-01 hasta 1955-09-19.

La variedad y tipo de formatos de los documentos originales requirió un extenso trabajo de manipulación, limpieza y ordenamiento de los datos.

## 6.2. Herramientas Utilizadas

Para el procesamiento de los datos se utilizó una extensión del lenguaje R[1] y del ecosistema de librerías Tidyverse[2], un conjunto paquetes diseñados para ciencia de datos que comparten una misma filosofía de diseño, gramática y estructura de datos. Además, se utilizó RMarkdown[3], un tipo de documento que permite combinar texto narrado con código de distintos lenguajes, incluido R, Python y SQL y exportar los contenidos en forma de reportes, libros u otros en HTML, PDF o Word.

Para facilitar la reproducibilidad, en el Anexo 10.1 se adjunta información de la sesión de trabajo con la versión de todas las librerías y herramientas utilizadas.

Específicamente para la parte de Procesamiento de Lenguaje Natural se utilizó el módulo Spacy[4], una librería de código abierto para NLP avanzado, escrita en Python y Cython, que soporta más de 64 lenguajes humanos. En particular se utilizó es_core_news_sm[5], un modelo en español pre-entrenado en los corpus AnCora[6] y WikiNER[7]. AnCora está formado por un corpus de 500 mil palabras provenientes principalmente de artículos periodísticos, con información morfológica, sintáctica, semántica y pragmática. WikiNER es un corpus que surge de explotar el texto en diferentes lenguajes humanos y la estructura de Wikipedia.

## 6.3. Extracción de los Datos

Los documentos originales conteniendo los discursos nos fueron proporcionados en formato digital por el mismo Enrique de Alzáa. En el lote recibido había documentos con cuatro formatos distintos: .DOC, .doc, .rtf y .PDF .

[1] https://www.r-project.org/
[2] https://www.tidyverse.org/
[3] https://rmarkdown.rstudio.com/
[4] https://spacy.io/
[5] https://spacy.io/models/es#es_core_news_sm
[6] https://github.com/UniversalDependencies/UD_Spanish-AnCora
[7] Learning multilingual named entity recognition from Wikipedia (figshare.com)

Para leer los documentos según el formato que correspondía, se utilizaron las librerías de R readtext[8], textreadr[9] y pdftools[10].

En el Anexo 10.2 se encuentra el código fuente, con los procesos aplicados.

## 6.4. Proceso ETL Aplicado

Inicialmente, aplicamos sobre los documentos un proceso de limpieza semiautomático, que requiere poca intervención humana.

Para tareas de limpieza y manipulación de texto previo al análisis se utilizó la librería stringr[11], una librería de R para la manipulación de cadenas de caracteres, combinada con un uso extensivo de expresiones regulares. Para la extracción de las fechas de los discursos se utilizó la librería lubridate[12].

Extraemos la fecha y el título de cada documento, como variables análisis. En el caso de los títulos, eliminamos todos los fin-de-línea del final y reemplazamos por un blanco los que se encuentre en el medio. También eliminamos los espacios en blanco que rodean los títulos, cuando amerita.

Una vez extraídos fecha y título, se eliminan del documento, para que sólo quede el texto propiamente dcho. Dentro de cada texto, reemplazamos los fin-de-línea por espacios simples y luego reincorporamos los punto-y-aparte. El texto puro pasa a ser la tercera variable de análisis.

Si bien todo este proceso es análogo en todos los documentos, en el Anexo 10.2 se puede consultar el código que se programó para cada tipo de documento, ya que los documentos .doc, no se comportan igual que los .DOC (que son Rich Text Format), ni los RTF o PDF. Sin extendernos en este aspecto técnico, a modo de simple ejemplo, los RTF tienen '\n' en lugar de '\r\n' para señalar el fin-de-línea.

[8] https://rdrr.io/cran/readtext/src/R/readtext.R
[9] https://rdrr.io/cran/textreadr/
[10] https://rdrr.io/cran/pdftools/
[11] https://stringr.tidyverse.org/
[12] https://lubridate.tidyverse.org/

En el caso de los PDF, hay un tratamiento distinto. La fecha y el título se pueden extraer fácilmente del índice. Para el texto, debimos eliminar los pie-de-página y luegolos tramos de los pie-de-página que se extendieron a la página siguiente, como así también las palabras cortadas, los espacios sin sentido y los números de página.

En cada proceso, transformamos fecha al formato correspondiente.

Finalmente, unificamos todos los dataframes que fuimos obteniendo en uno solo.

En el Anexo 10.2 se encuentra el código fuente con los procesos aplicados.

El resultado de todo este proceso de ETL es un dataframe peron_df con tres variables: fecha, titulo y texto. En la Fig. 1 se puede ver un fragmento del mencionado dataframe, con su formato.

| | fecha | titulo | text |
|---|---|---|---|
| 1 | 1943-12-01 | Declaraciones del coronel Perón luego haber sido designa... | Los patrones, los obreros y el Estado constituyen las parte... |
| 2 | 1943-12-02 | Al asumir el cargo de secretario de Trabajo y Previsión : | Excelentísimo señor presidente: ... |
| 3 | 1943-12-02 | Discurso luego de asumir el cargo de secretario de Trabajo ... | En el tiempo que estuve al frente del ex Departamento ... |
| 4 | 1944-01-16 | Mensaje para informar sobre las tareas para ayudar a las ... | Nos dirigimos a todo el pueblo de la patria, en nombre del ... |
| 5 | 1944-04-04 | Al hacer entrega de los fondos de la coleta Pro-Víctimas de ... | Excelentísimo Señor Presidente de la Nación: ... |
| 6 | 1944-05-29 | En el almuerzo realizado en la Fábrica Militar de Aviones de ... | El bautismo de este nuevo avión, que se incorpora a las fue... |

Fig. 1 – Fragmento del dataframe peron.df

## 6.5. Procesamiento de Lenguaje Natural

Como se mencionó anteriormente, para esta sección se utilizó la librería Spacy de Python. El procesamiento consiste en tokenización, lematización, etiquetado de partes de discurso (part-of-speech tagging) y reconocimiento de entidades (named entity recognition).

A continuación, explicaremos brevemente en qué consiste cada proceso:

- Tokenización: es el proceso de segmentación de un texto en unidades menores, típicamente en palabras.
- Lematización: por razones gramaticales los documentos utilizan diferentes formas de una palabra (organizar, organiza, organizando). Lematización es un proceso que permite reducir las formas derivadas de una palabra asociando cada forma derivada con la palabra base o lema.
- Part-of-speech tagging: es el proceso de asignar a cada palabra en un documento su categoría gramatical de acuerdo tanto a su definición como al contexto en el que es utilizada.
- Named Entity Recognition: es el proceso que busca identificar categorías pre-definidas como personas, organizaciones, lugares, etc. Dentro de un texto.

Se puede consultar el código fuente en el Anexo 10.3.

El resultado de estos procesos son dos dataframes. El primero contiene los resultados de la tokenización, lematización y etiquetado de partes del discurso. En la Figura 2 se muestran las primeras filas de dicho dataframe.

El segundo dataframe corresponde al reconocedor de entidades. El modelo en español utilizado identifica cuatro tipo de entidades: lugares (LOC), organizaciones (ORG), personas (PER) y otros (MISC). En la Figura 3 se muestran las primeras filas obtenidas.

Utilizar un modelo pre-entrenado con corpus en Español es una posibilidad relativamente reciente, ya que los desarrollos en NLP suelen hacerse con corpus en inglés y representa una ventaja evidente frente a tener que traducir el texto en español que se quiere analizar o utilizar corpus “adaptados”.

Sin embargo, la disponibilidad de corpus en Español aún es acotada y es difícil encontrar corpus específicos sobre el tema del texto que uno quiere analizar. Además, muchas veces los modelos en Español tienen menos funcionalidades y categorías que los modelos en inglés.

| | text | lemma | is_punctuation | is_space | shape | part_of_speech | pos_tag | id |
|---|---|---|---|---|---|---|---|---|
| 1 | Los | Los | FALSE | FALSE | Xxx | DET | DET__Definite=Def\|Gender=Masc\|Number=Plur\|PronType=... | 0 |
| 2 | patrones | patrón | FALSE | FALSE | xxxx | NOUN | NOUN__Gender=Masc\|Number=Plur | 0 |
| 3 | , | , | TRUE | FALSE | , | PUNCT | PUNCT__PunctType=Comm | 0 |
| 4 | los | lo | FALSE | FALSE | xxx | DET | DET__Definite=Def\|Gender=Masc\|Number=Plur\|PronType=... | 0 |
| 5 | obreros | obrero | FALSE | FALSE | xxxx | NOUN | NOUN__Gender=Masc\|Number=Plur | 0 |
| 6 | y | y | FALSE | FALSE | x | CCONJ | CCONJ | 0 |
| 7 | el | el | FALSE | FALSE | xx | DET | DET__Definite=Def\|Gender=Masc\|Number=Sing\|PronType=... | 0 |
| 8 | Estado | Estado | FALSE | FALSE | Xxxxx | PROPN | PROPN | 0 |
| 9 | *NA* | *NA* | FALSE | TRUE | *NA* | SPACE | _SP | 0 |
| 10 | constituyen | constituir | FALSE | FALSE | xxxx | AUX | VERB__Mood=Ind\|Number=Plur\|Person=3\|Tense=Pres\|Verb... | 0 |
| 11 | *NA* | *NA* | FALSE | TRUE | *NA* | SPACE | _SP | 0 |
| 12 | las | los | FALSE | FALSE | xxx | DET | DET__Definite=Def\|Gender=Fem\|Number=Plur\|PronType=Art | 0 |
| 13 | *NA* | *NA* | FALSE | TRUE | *NA* | SPACE | _SP | 0 |
| 14 | partes | partir | FALSE | FALSE | xxxx | NOUN | NOUN__Gender=Fem\|Number=Plur | 0 |
| 15 | *NA* | *NA* | FALSE | TRUE | *NA* | SPACE | _SP | 0 |
| 16 | de | de | FALSE | FALSE | xx | ADP | ADP__AdpType=Prep | 0 |
| 17 | *NA* | *NA* | FALSE | TRUE | *NA* | SPACE | _SP | 0 |
| 18 | todo | todo | FALSE | FALSE | xxxx | DET | DET__Gender=Masc\|Number=Sing\|PronType=Tot | 0 |
| 19 | problema | problema | FALSE | FALSE | xxxx | NOUN | NOUN__Gender=Masc\|Number=Sing | 0 |
| 20 | social | social | FALSE | FALSE | xxxx | ADJ | ADJ__Number=Sing | 0 |

Fig. 2 – Fragmento del dataframe con tokenización, lematización y etiquetado de partes del discurso

En el caso de este trabajo, como se mencionó anteriormente, el corpus AnCora está basado mayoritariamente en artículos periodísticos por lo que se espera cierta pérdida de precisión en el análisis de discursos políticos o gubernamentales. En el caso de Reconocimiento de Entiedades Nombradas (NER) el modelo en Español utilizado en este trabajo reconoce cuatro categorías de entidades, mientras que los modelos pre-entrenados en inglés de la misma librería Spacy reconocen 18 categorías.

| | ent | label | id |
|---|---|---|---|
| 1 | Estado | MISC | 0 |
| 2 | Estado argentino | LOC | 0 |
| 3 | Todo conflicto | MISC | 0 |
| 4 | En este sentido | MISC | 0 |
| 5 | Estado | MISC | 0 |
| 6 | Secretaría de Trabajo | ORG | 0 |
| 7 | Previsión | ORG | 0 |
| 8 | Estado | MISC | 0 |
| 9 | Estado | PER | 0 |
| 10 | Por su parte | MISC | 0 |
| 11 | En tal sentido | MISC | 0 |
| 12 | La oportunidad de las reformas | MISC | 0 |
| 13 | La revisión de los textos | MISC | 0 |
| 14 | Secretaría | MISC | 0 |
| 15 | Es impropio | MISC | 0 |
| 16 | Derecho del Trabajo | MISC | 0 |
| 17 | Estado | LOC | 0 |
| 18 | Lo que al respecto haga la nueva organización | MISC | 0 |
| 19 | En mérito | MISC | 0 |
| 20 | Consejo Superior de Trabajo | MISC | 0 |

Fig. 3 – Fragmento del dataframe con reconocimiento de entidades

# 7. Análisis de los Datos

Con la aplicación del proceso de Extracción y Procesamiento descripto en la Sección 6, hemos obtenido un dataframe unificado que contiene los 1163 discursos emitidos entre 1943-12-01 y 1955-09-19,

En la presente Sección, inspeccionamos dicho dataframe con el fin de dar respuesta a las cuestiones planteadas en los objetivos.

## 7.1. Análisis Descriptivo de los Discursos

Comenzamos analizando la frecuencia de los discursos a través del tiempo, utilizando la siguiente consulta:

```
peron_df <- read_csv("peron_discursos_completos.csv")

peron_df %>%
  mutate(anio = year(fecha)) %>%
  count(anio) %>%
  ggplot(aes(factor(anio), y = n)) +
  geom_col(fill="lightblue") +
  theme_bw() +
  labs(x = "Año",
       y = "Cantidad de discursos") +
  geom_hline(aes(yintercept = mean(n), color = "Media")) +
  geom_hline(aes(yintercept = median(n), color = "Mediana")) +
  theme(legend.title = element_blank())
```

En la Figura 4 se puede ver el resultado en forma gráfica.

De este análisis se desprende que, en promedio, Perón pronunció 89.5 por año. Considerando que hay dos años incompletos, 1943 y 1955, podemos tomar una medida robusta a valores atípicos como la mediana. La misma es de 99.

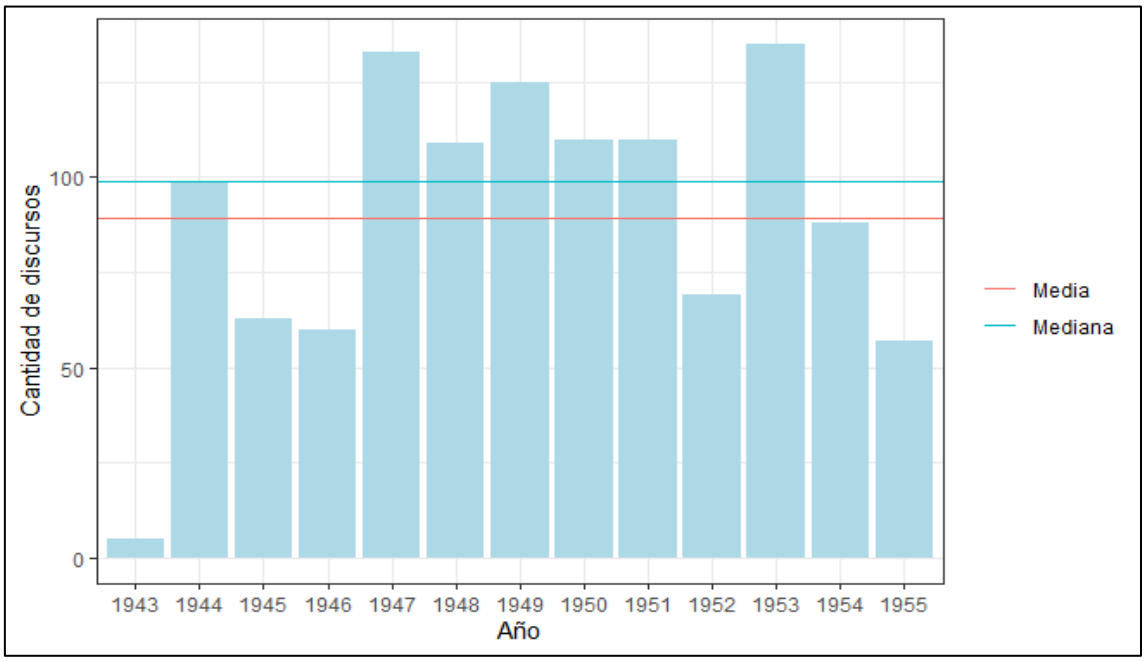


Fig. 4 – Periodicidad de los discursos a través del tiempo

Un análisis más profundo puede detectar si hubo más discursos durante los años de elecciones:

```
peron_df %>%
  mutate(anio = year(fecha),
         elecciones = case_when(year(fecha) == 1946 ~ "Presidenciales",
                                year(fecha) == 1951 ~ "Presidenciales",
                                year(fecha) == 1948 ~ "Convencionales Constit
uyentes",
                                year(fecha) == 1954 ~ "De Vicepresidente",
                                TRUE ~ "Sin elecciones")) %>%
  ggplot(aes(factor(anio),
             fill = factor(elecciones,
                           levels = c("Sin elecciones",
                                      "Presidenciales",
                                      "Convencionales Constituyentes",
                                      "De Vicepresidente")))) +
  geom_bar() +
  theme_bw() +
  labs(x = "Año",
       y = "Cantidad de discursos") +
  scale_fill_discrete(name = "Elecciones") +
  theme(legend.position = "bottom")
```

En la Figura 5 podemos ver gráficamente este resultado.

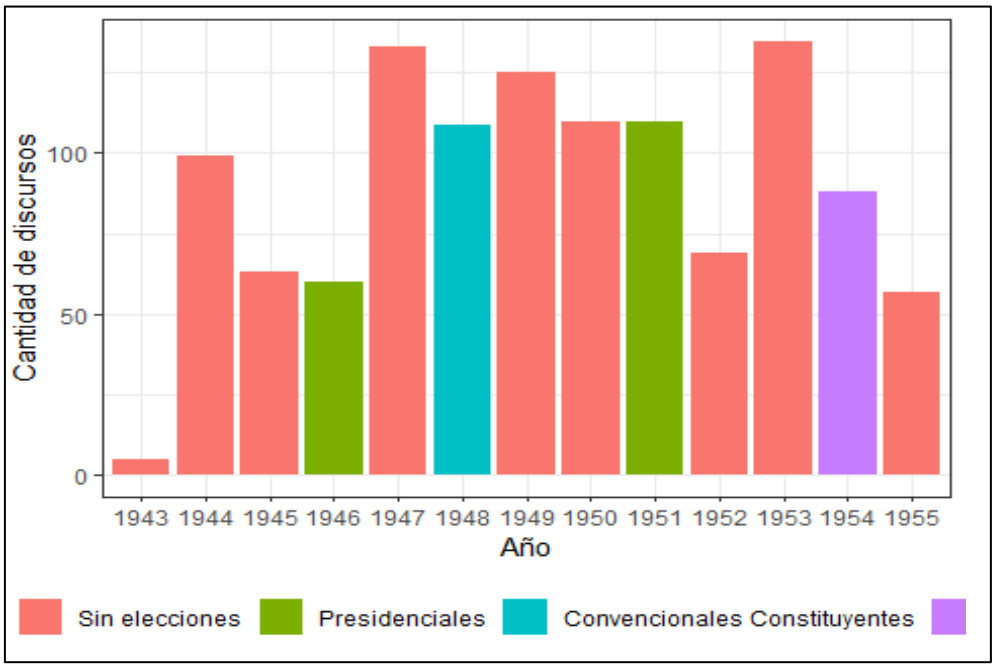


Fig. 5 – Periodicidad de los discursos discriminados por año electoral.

A simple vista podríamos concluir que en los años de elecciones hubo menos discursos. Sin embargo, este análisis puede estar distorsionado si las elecciones se deban al comienzo del año, como las presidenciales de 1946 que se celebraron el 24 de febrero.

Refinamos el análisis a través de la distribución de discursos a lo largo de los meses:

```
peron_df %>%
  mutate(anio = year(fecha),
         mes = month(fecha)) %>%
  count(anio, mes) %>%
  group_by(mes) %>%
  summarise(media_mes = mean(n)) %>%
  ggplot(aes(factor(mes), media_mes)) +
  geom_col(fill = "lightblue") +
  theme_bw() +
  labs(x = "Mes",
       y = "Media")
```

Del resultado, podemos concluir que se destaca el mes de octubre como el mes en que más discursos realizaba Perón. En la Figura 6 podemos ver el resultado en forma visual.

Un análisis sociológico, que excede al presente trabajo, podría identificar la causa de este comportamiento observable (acontecimientos políticos de contexto, cercanía de eventos personales, etc.),

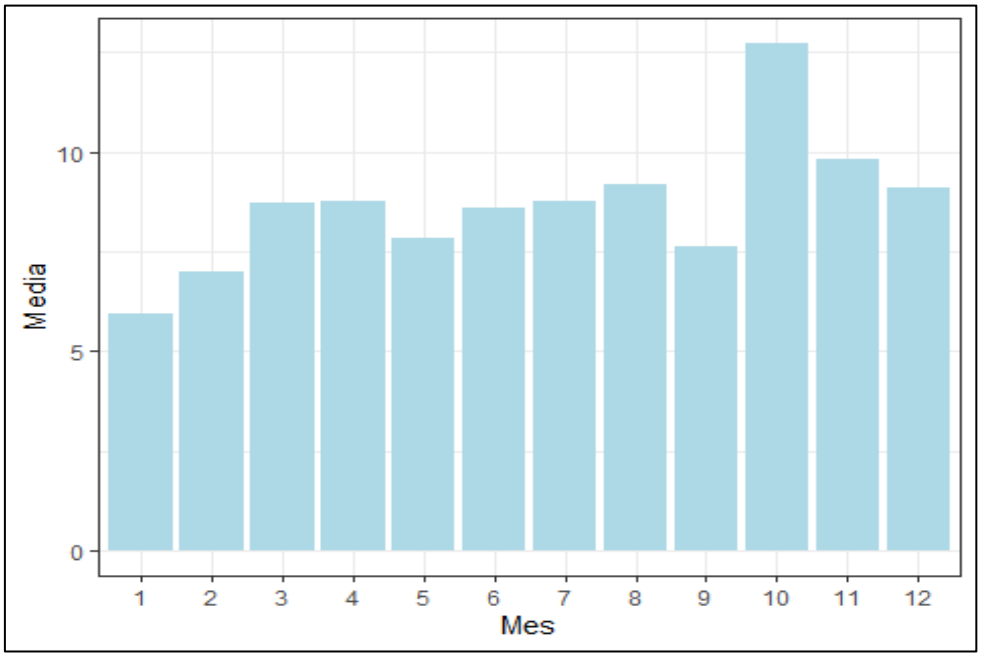


Fig. 6 – Periodicidad de los discursos discriminados por mes.

Ahora vamos a estudiar la extensión de los discursos:

```
peron_meta <- read_csv("peron_discursos_completos.csv")
peron_meta <- peron_meta %>%
  select(-text) %>%
  mutate(Año = lubridate::year(fecha))

peron_annot_docs_df %>%
  group_by(id) %>%
  summarise(Cantidad = n()) %>%
  bind_cols(peron_meta) %>%
  ggplot(aes(fecha, Cantidad)) +
  geom_line(color = grey(0.8)) +
  geom_point(color = "salmon") +
  geom_smooth(method = "loess", formula = y ~ x) +
  scale_x_date(date_labels =  "%Y", date_breaks = "1 year") +
  theme_bw() +
  labs(x = "Fecha")
```

En la Fig. 7 se puede visualizar la distribución obtenida. Encontramos un valor atípico, que excede el umbral de la extensión del resto de los discursos, y que se sitúa en el año 1951.

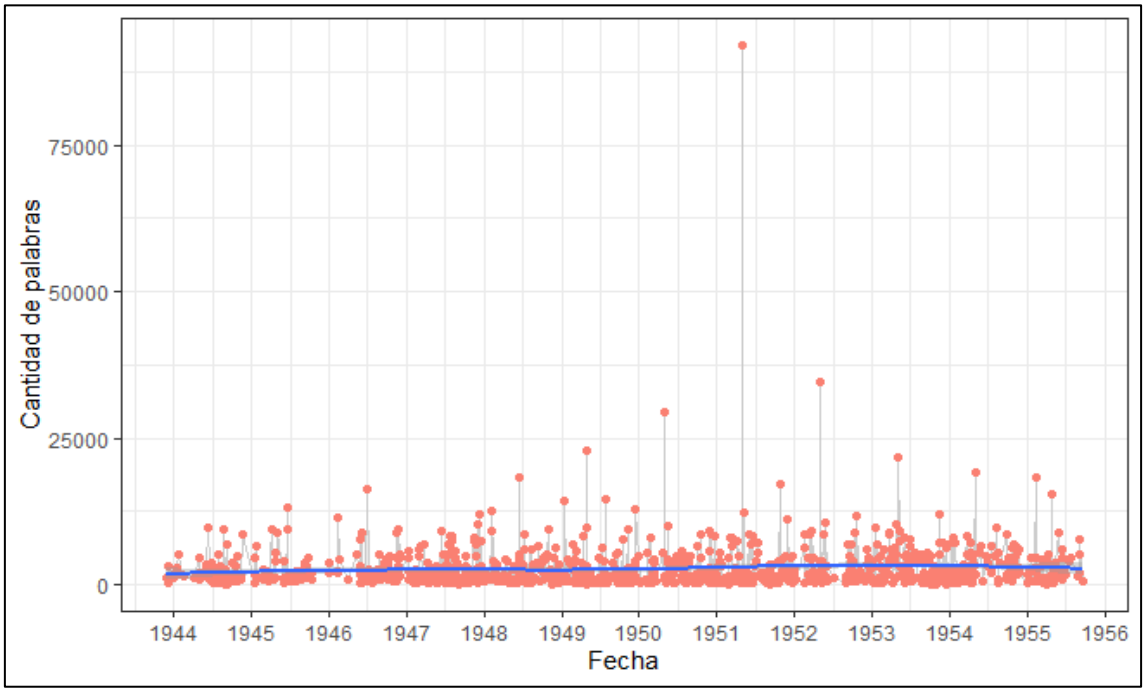


Fig. 7 – Distribución de los discursos por su extensión.

Realizamos un análisis puntual del dato atípico en 1951, y detectamos que se trató del discurso de Apertura de las Sesiones Legislativas del 1ro de Mayo de 1951:

```
peron_annot_docs_df %>%
  group_by(id) %>%
  summarise(Cantidad = n()) %>%
  bind_cols(peron_meta) %>%
  arrange(desc(Cantidad)) %>%
  top_n(1, Cantidad)

## # A tibble: 1 x 5
##      id Cantidad fecha      titulo
Año
##   <dbl>    <int> <date>     <chr>
<dbl>
## 1   202    93708 1951-05-01 ASAMBLEA LEGISLATIVA                    ~  1951
```

Dado que este discurso podría considerare como “outlier”, lo excluimos y repetimos el análisis d extensión sobre el resto de los discursos:

```
peron_annot_docs_df %>%
  group_by(id) %>%
  summarise(Cantidad = n()) %>%
  bind_cols(peron_meta) %>%
  filter(id != 202) %>%
  ggplot(aes(fecha, Cantidad)) +
  geom_line(color = grey(0.8)) +
  geom_point(color = "salmon") +
  geom_smooth(method = "loess", formula = y ~ x) +
  theme_bw() +
  scale_x_date(date_breaks = "1 year", date_labels = "%Y") +
  labs(x = "Fecha")
```

El resultado de este último análisis puede verse en la Figura 8.

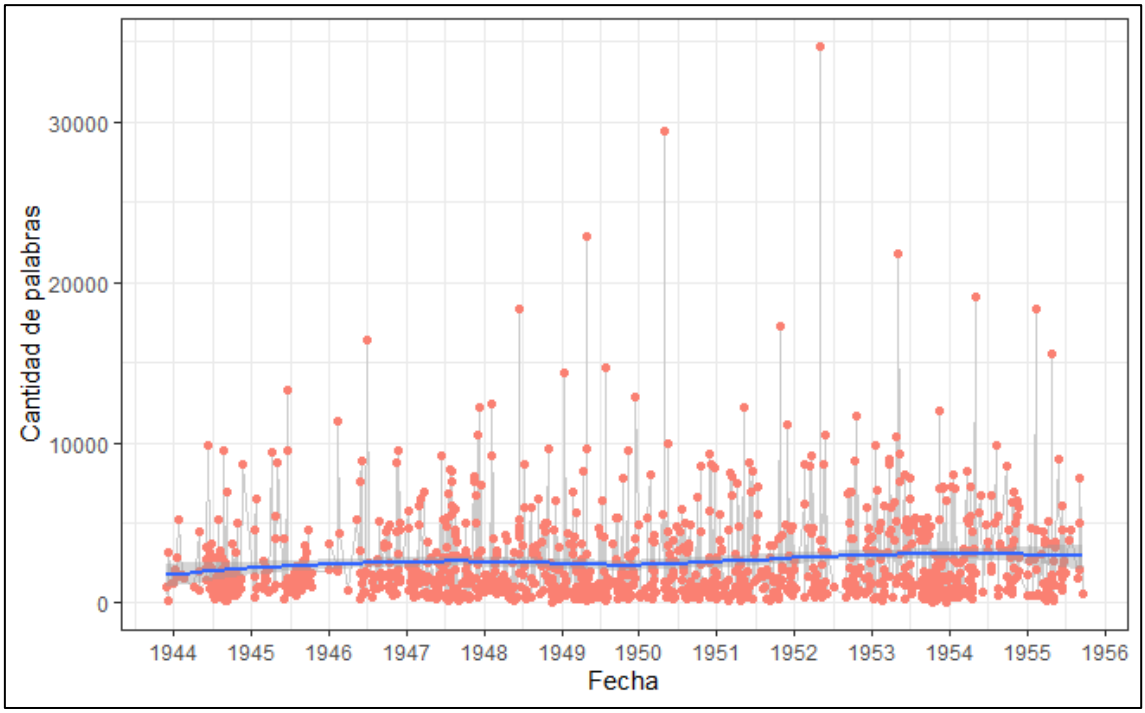


Fig. 8 – Distribución de los discursos por su extensión, excluyendo la Apertura de las Sesiones Legislativas del 1ro de Mayo de 1951

Dado que no se encuentra en esta distribución patrón evidente a lo largo del tiempo, repetimos el último análisis en forma independientemente del tiempo:

```
peron_annot_docs_df_sum <- peron_annot_docs_df %>%
  group_by(id) %>%
  summarise(Cantidad = n()) %>%
  summarise(Media = mean(Cantidad), Mediana = median(Cantidad))

peron_annot_docs_df %>%
  group_by(id) %>%
  summarise(Cantidad = n()) %>%
  filter(id != 202) %>%
  ggplot() +
  geom_histogram(aes(x=Cantidad),fill = "salmon", binwidth = 1000) +
  scale_x_continuous(breaks = seq(0,35000,1000)) +
  geom_vline(aes(xintercept = Media, color = "Media"), peron_annot_docs_df_su
m) +
  geom_vline(aes(xintercept = Mediana, color = "Mediana"), peron_annot_docs_d
f_sum) +
  scale_color_manual(name = "Medida", values = c(Media = "black", Mediana = "
blue")) +
  theme_bw() +
  labs(title = "Distribución de la extensión de los discursos",
       x = "Extensión", y = "Cant. de discuros") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0))
```

En la Figura 9 encontramos el resultado del análisis en forma gráfica.

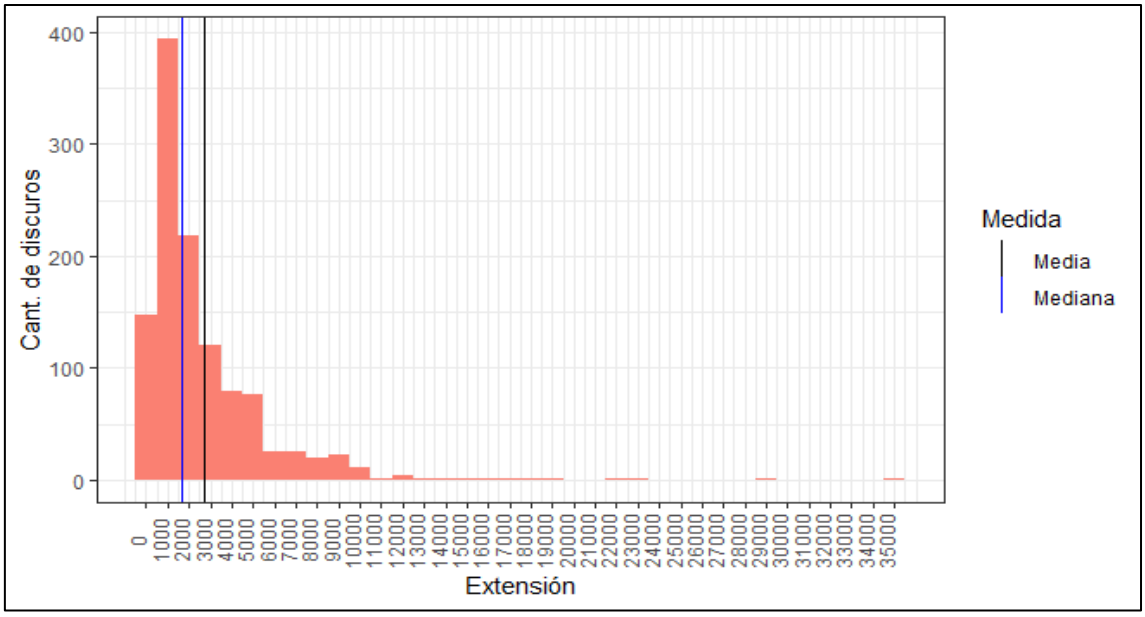


Fig. 9 – Distribución de los discursos por su extensión con independencia del tiempo

De este resultado podemos concluir que los discursos tienen una distribución asimétrica hacia la derecha, con una mediana de aproximadamente 1.600 palabras y algunos pocos discursos excepcionalmente largos, como el mencionado de la Apertura de Sesiones de 1951, la primera de su segundo mandato. Además, concluimos que la extensión de los discursos no varía a lo largo del tiempo.

## 7.2. Análisis exploratorio del contenido de los discursos

En esta sección hacemos un análisis sobre el contenido textual de los discursos.

Inicialmente, tratamos de identificar tendencia de los títulos de los discursos, mediante Nube de Palabras:

```
library(tm)
library(wordcloud)

titulos <- select(peron_df, -text)

stop_words <- tibble(word = tm::stopwords("spanish"),
                     lexicon = "tm package")
titulos <- titulos %>%
  unnest_tokens(word, titulo) %>%
  anti_join(stop_words)

titulos %>%
  count(word, sort = TRUE) %>%
  {wordcloud(words = .$word,
             freq = .$n,
             max.words = 200,
             random.order = FALSE,
             rot.per = .35,
             colors=brewer.pal(8, "Dark2"))}
```

En la Figura 10 podemos visualizar la nube obtenida.


Fig. 10 – Nube de palabras sobre los títulos de los discursos.

Buscaremos a continuación, la distribución de las 25 palabras más frecuentes en los títulos:

```
titulos %>%
  count(word, sort = TRUE) %>%
  head(25) %>%
  ggplot(aes(fct_reorder(word, n), n)) +
  geom_col(fill = "lightsalmon") +
  coord_flip() +
  theme_bw() +
  labs(x = "Palabras",
       y = "Frecuencias")
```

En la Figura 11 se puede visualizar la distribución obtenida de las 25 palabras más frecuentes en los títulos. Más allá de palabras genéricas como “acto” o “inauguración” hay una clara preponderancia en los títulos a referencias sindicales a través de palabras como “trabajo”, “delegados”, “obreros”.

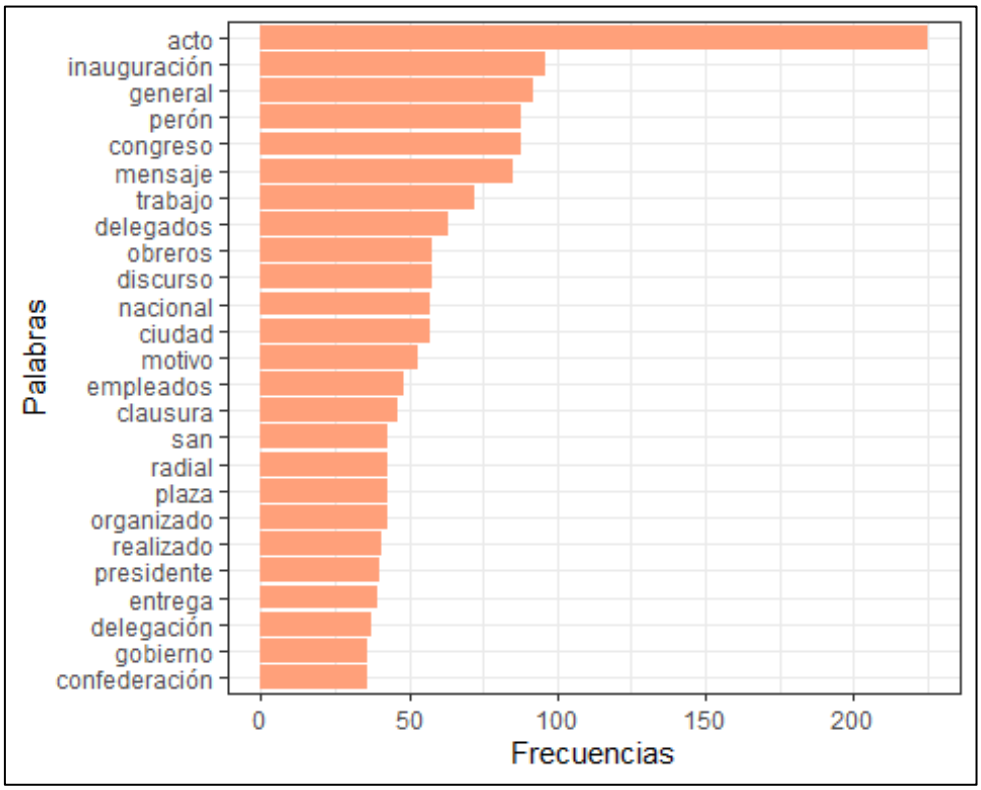


Fig. 11 – Distribución de las 25 palabras más frecuentes en los títulos

Finalmente, repetiremos el análisis anterior sobre el contenido de los discursos, con una primera aproximación a través de los sustantivos:

```
peron_annot_docs_df %>%
  filter(part_of_speech == "NOUN") %>%
  group_by(lemma) %>%
  summarise(Cantidad = n()) %>%
  top_n(20, Cantidad) %>%
  arrange(desc(Cantidad)) %>%
  ggplot(aes(fct_reorder(lemma, Cantidad), Cantidad)) +
  coord_flip() +
  geom_col(fill = "salmon") +
  theme_bw() +
  labs(title = "Principales sustantivos",
       x = "lemma")
```

En la Figura 12 se puede visualizar la distribución obtenida de las 20 palabras más frecuentes en los textos de los discursos.

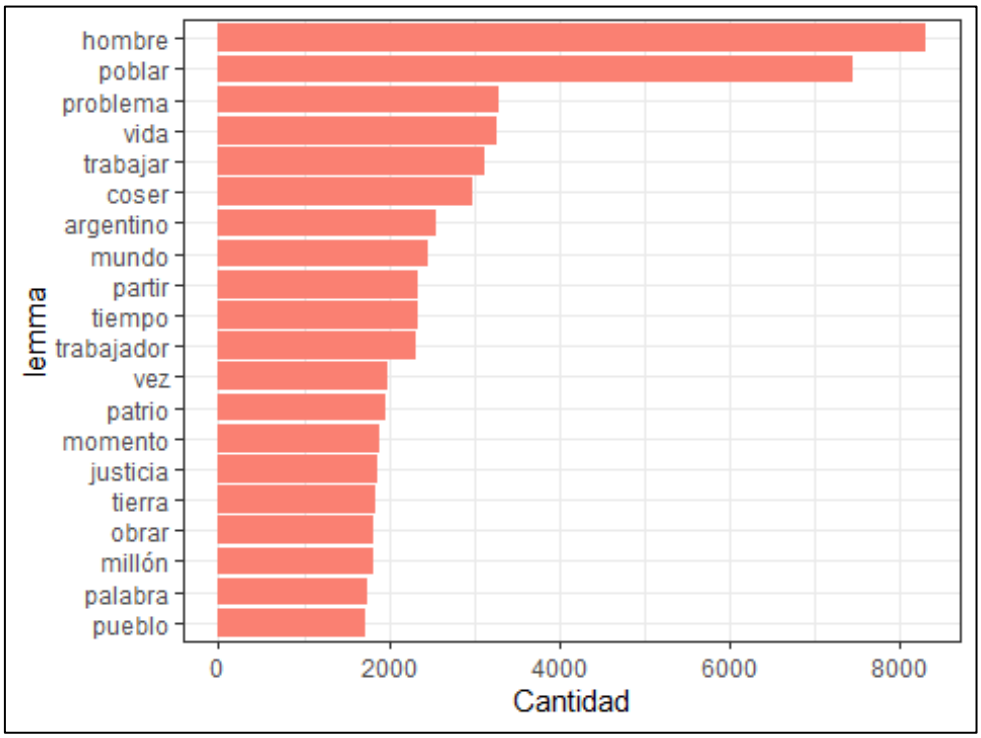


Fig. 12 – Distribución de las 20 palabras más frecuentes en los textos de los discursos

Una lectura inicial de los principales sustantivos permite formular unas primeras consideraciones y preguntas para profundizar luego en una lectura más detenida de los textos.

En primer lugar, el sustantivo más repetido, "hombre", hace referencia seguramente a "persona", lejos de las consideraciones de género actuales que las décadas de 1940 y 1950 no existían.

En segundo lugar, se pueden identificar un grupo de palabras relacionadas con el mundo del trabajo o sindical, como son "organización", "trabajar", "trabajador", "compañero" o "pueblo". También se puede identificar un grupo de palabras que pueden estar relacionadas con la cosmovisión nacionalista, como ser "poblar", "argentino" o "patrio".

Buscaremos ahora, la "cobertura" de estas 20 principales palabras, es decir, la cantidad de discursos en las que aparece cada una de ellas:

```
peron_annot_docs_df %>%
  filter(part_of_speech == "NOUN") %>%
  count(lemma, sort = TRUE, name = "Cantidad") %>%
  top_n(20, Cantidad) %>%
  {.$lemma} -> peron_sust20

peron_annot_docs_df %>%
  filter(lemma %in% peron_sust20) %>%
  count(lemma, id) %>%
  count(lemma, sort = TRUE, name = "cantidad_de_discursos") %>%
  ggplot(aes(x = fct_reorder(lemma, cantidad_de_discursos),
             y = cantidad_de_discursos)) +
  coord_flip() +
  geom_col(fill = "salmon") +
  theme_bw() +
  labs(title = "Cobertura de las principales palabras",
       x = "lemma", y = "Cantidad de discursos")
```

El resultado lo podemos apreciar en la Figura 13.

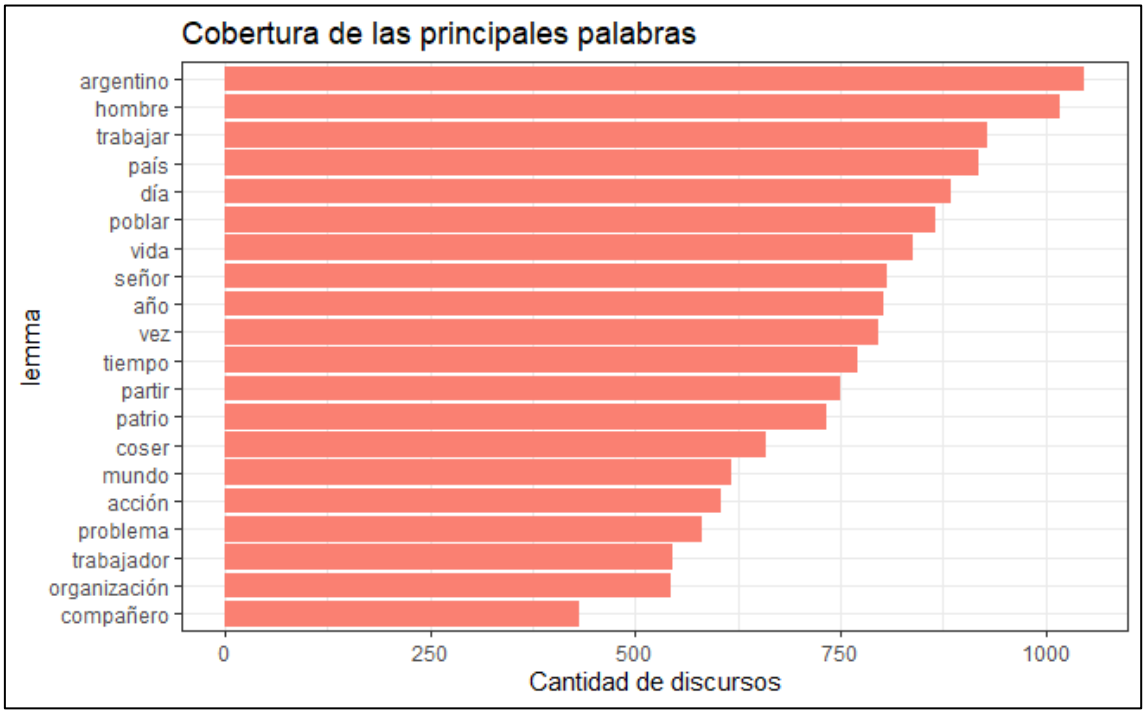


Fig. 13 - Cobertura de los 20 principales sustantivos

Teniendo en cuenta que estamos analizando un universo de 1.163 discursos casi todas las palabras aparecen en aproximadamente la mitad o más de los discursos. Esto refuerza la idea que son conceptos que se mantuvieron a lo largo del período analizado.

Veamos ahora los principales verbos, excluyendo algunos muy genéricos como “ser”, “estar” y “haber”:

```
peron_annot_docs_df %>%
  filter(part_of_speech == "VERB" & !(lemma %in% c("ser", "estar", "haber")))
%>%
  count(lemma, sort = TRUE, name = "Cantidad") %>%
  top_n(20, Cantidad) %>%
  ggplot(aes(fct_reorder(lemma, Cantidad), Cantidad)) +
  coord_flip() +
  geom_col(fill = "salmon") +
  theme_bw() +
  labs(title = "Principales verbos",
       x = "lemma")
```

En la Figura 14 podemos consultar el resultado obtenido. Tomados aisladamente, no parece poderse extraer ninguna conclusión del análisis de los verbos.

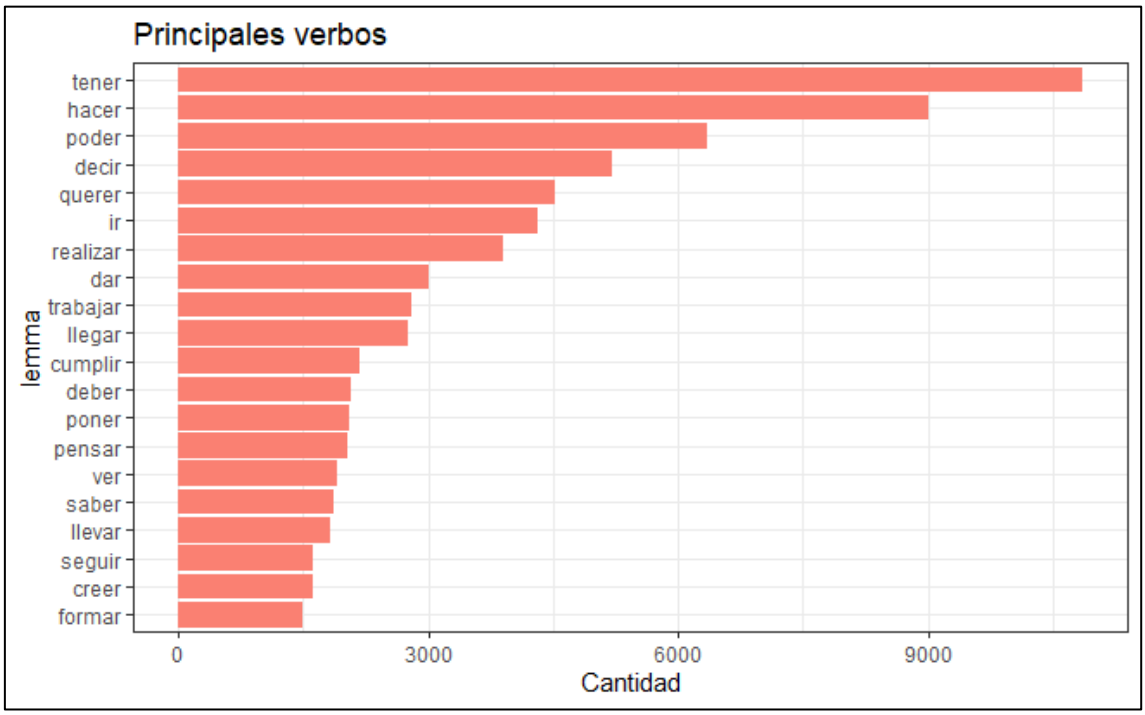


Fig. 14 – Principales verbos utilizados

Analizaremos ahora las Entidades reconocidas. Como se explicó anteriormente, nuestro modelo reconoce cuatro categorías de entidades:

- PER: nombres propios o comunes
- ORG: empresas, agencias, instituciones, etc.
- LOC: tanto nombres propios de lugares, como nombres comunes (por ej. montañas, lagos)
- MISC: misceláneo se refiere a entidades diversas como pueden ser eventos, nacionalidades, religiones, productos, obras de arte, etc.

En primer lugar, analizaremos las 20 principales entidades de las 4 categorías en un mismo gráfico y luego analizaremos las 20 principales de cada categoría. En todos los casos se filtraron algunas palabras mal clasificadas o aquellas que son tan genéricas que no aportan sentido al análisis:

```
peron_annot_ents_df %>%
  filter(!(ent %in% c("Eso", "Por eso", "Por esa razón", "¿", "“", "Así",
"También"))) %>%
  count(ent, label, sort = TRUE, name = "Cantidad") %>%
  top_n(20, Cantidad) %>%
  ggplot(aes(fct_reorder(ent, Cantidad), Cantidad, fill = label)) +
  coord_flip() +
  geom_col() +
  theme_bw() +
  labs(title = "Principales entidades",
       x = "Entidad")
```

En la Figura 15 podemos ver el resultado gráficamente.

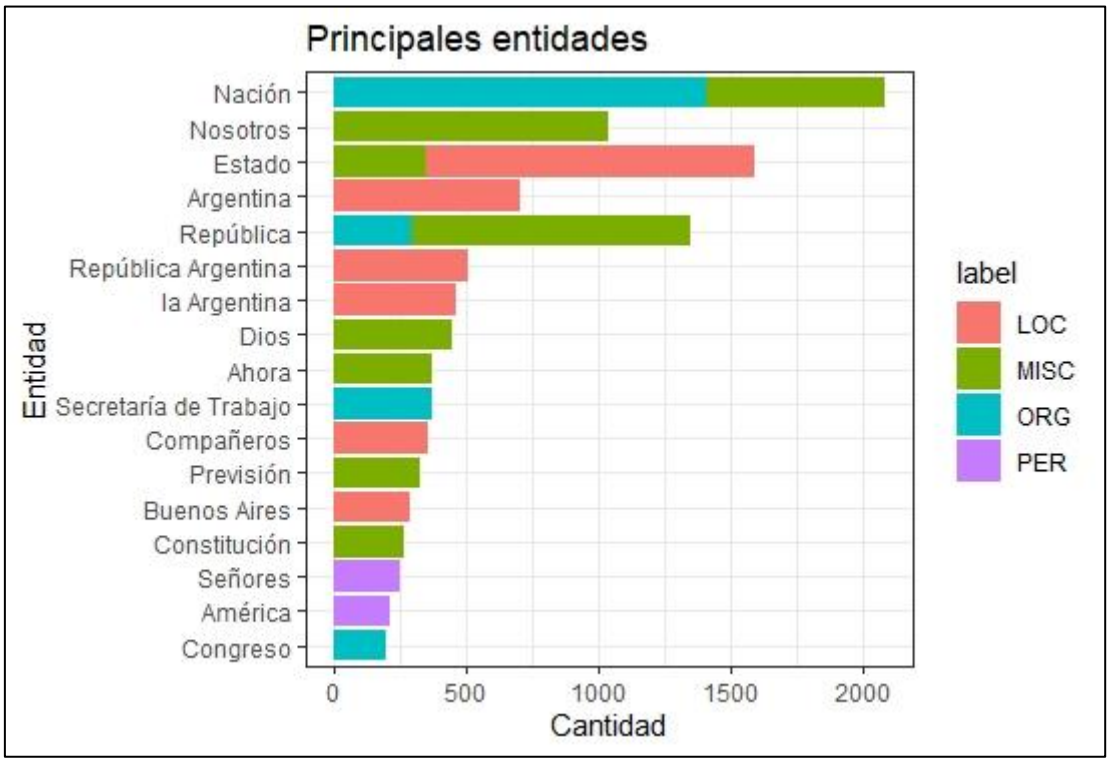


Fig. 15 - Las 20 principales entidades de las 4 categorías, PER, LOC, ORG y MISC

Finalmente, discriminaremos cada una de las cuatro categorías, para poder ver su comportamiento por separado. Los resultados pueden consultarse en las Figuras 16 a 19.

- *Principales personas:*

```
peron_annot_ents_df %>%
  filter(label == "PER" & !(ent %in% c("Aquí", "había", "Qué", "Después",
"Piensen", "Previsión", "Está", "Agradezco", "Había", "Eso", "Más", "…",
"¿Qué", "Puedo", "Naturalmente", "Quiero", "Deseo", "Recuerdo", "Sólo"))) %>%
  count(ent, sort = TRUE, name = "Cantidad") %>%
  top_n(20, Cantidad) %>%
  ggplot(aes(fct_reorder(ent, Cantidad), Cantidad)) +
  coord_flip() +
  geom_col(fill = "salmon") +
  theme_bw() +
  labs(title = "Principales personas",
       x = "Persona")
```

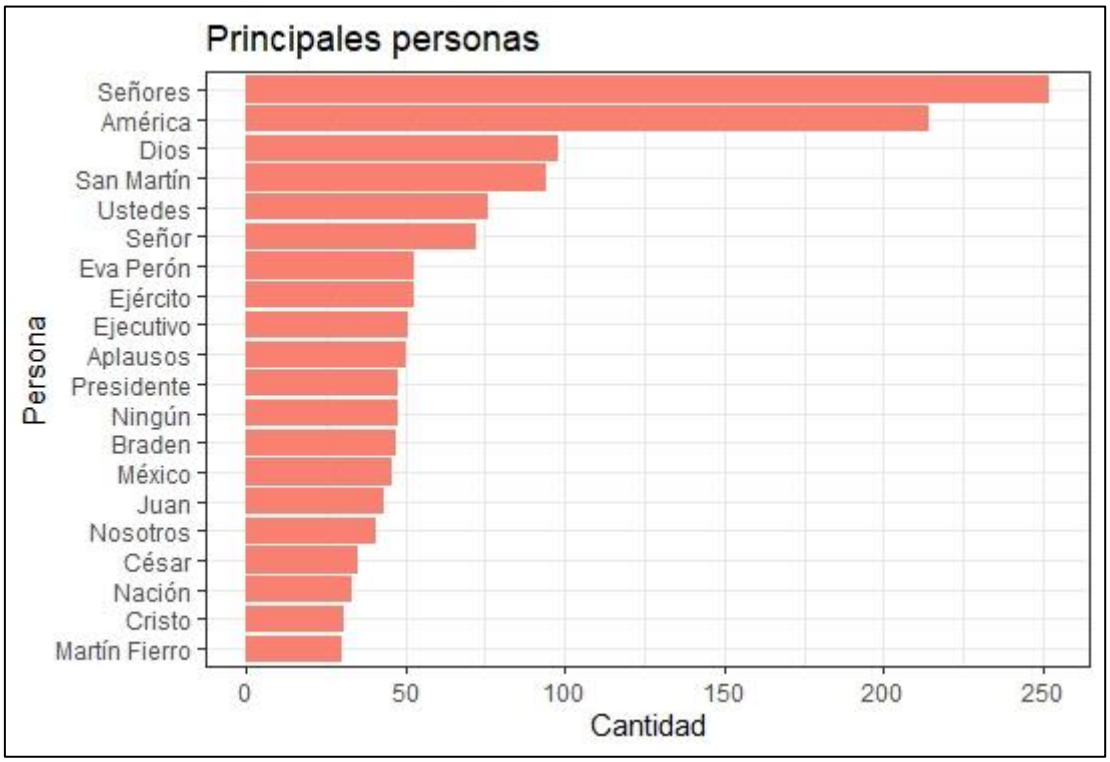


Fig. 16 - Las 20 principales entidades de categoría PER.

- *Principales organizaciones:*

```
peron_annot_ents_df %>%
  filter(label == "ORG" & !(ent %in% c("Afortunadamente"))) %>%
  count(ent, sort = TRUE, name = "Cantidad") %>%
  top_n(20, Cantidad) %>%
  ggplot(aes(fct_reorder(ent, Cantidad), Cantidad)) +
  coord_flip() +
  geom_col(fill = "salmon") +
  theme_bw() +
  labs(title = "Principales organizaciones",
       x = "Organización")
```


Fig. 17 - Las 20 principales entidades de categoría ORG.

- *Principales lugares:*

```
peron_annot_ents_df %>%
  filter(label == "LOC" & !(ent %in% c("Compañeros", "También", "Señores", "¿",
"Buenos", "Aires", "Cómo", "Después"))) %>%
  count(ent, sort = TRUE, name = "Cantidad") %>%
  top_n(20, Cantidad) %>%
  ggplot(aes(fct_reorder(ent, Cantidad), Cantidad)) +
  coord_flip() +
  geom_col(fill = "salmon") +
  theme_bw() +
  labs(title = "Principales lugares",
       x = "Lugar")
```

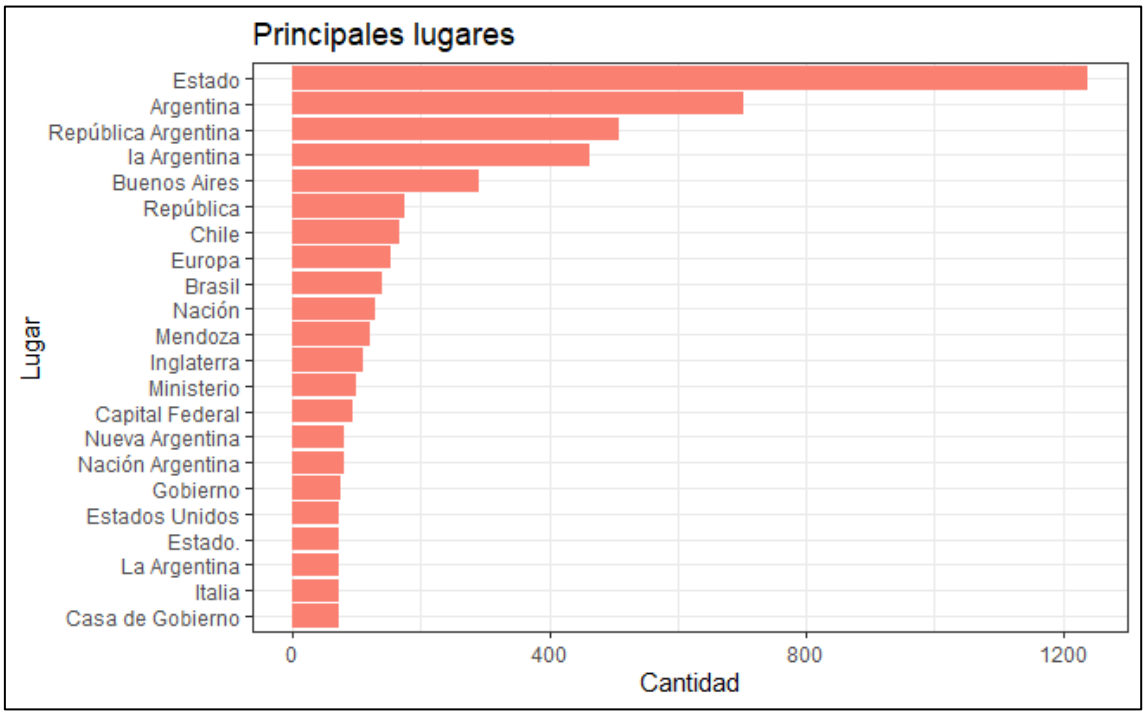


Fig. 18 - Las 20 principales entidades de categoría LOC.

- *Principales misceláneas:*

```
peron_annot_ents_df %>%
  filter(label == "MISC" & !(ent %in% c("Por eso", "Eso", "Ese", "¿", "Así",
"En", "Queremos", "Estamos", "Hemos", "No", "Y", "Por esa razón", "Sé", "Esa"
, "“", "Pero"))) %>%
  count(ent, sort = TRUE, name = "Cantidad") %>%
  top_n(20, Cantidad) %>%
  ggplot(aes(fct_reorder(ent, Cantidad), Cantidad)) +
  coord_flip() +
  geom_col(fill = "salmon") +
  theme_bw() +
  labs(title = "Principales MISC",
       x = "MISC")
```

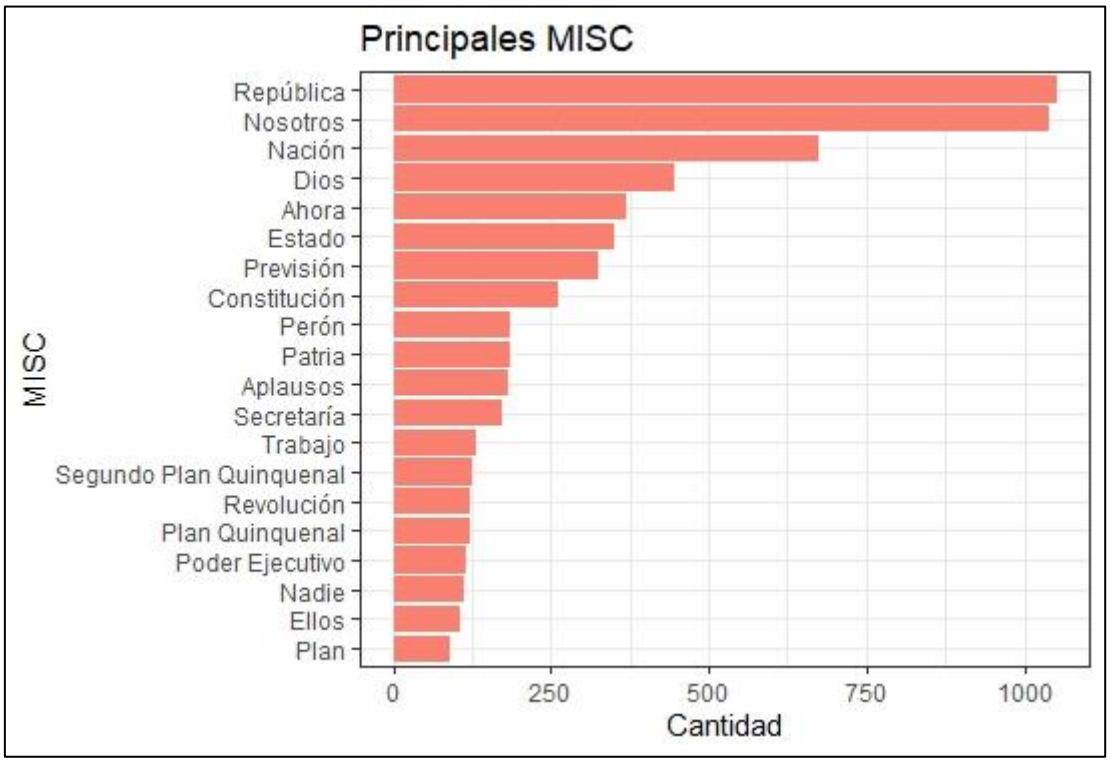


Fig. 19 - Las 20 principales entidades de categoría MISC.

Como se puede apreciar en una primera lectura de las entidades reconocidas en las 4 categorías, se observan conceptos muy significativos en el universo conceptual del peronismo en el período analizado. La repetida aparición de palabras como Nación y República, con alta frecuencia en todo el corpus, da cuenta de un estilo de movimiento.

En este sentido, se puede afirmar que esta herramienta puede ser útil en aportar pistas de cómo abordar el estudio de un período histórico o un movimiento político a través de los discursos de sus líderes.

# 8. Conclusiones y Trabajo Futuro

El análisis de discursos de un presidente ayuda a entender muchos aspectos del contexto político. En este trabajo se propuso un posible análisis de los discursos emitidos por el expresidente argentino Juan Domingo Perón entre 1943 y 1955, tanto desde el punto de vista descriptivo global como exploratorio de su contenido.

Se planteó un proceso de varias etapas, a través del cual se pudo obtener un corpus unificado sobre 1163 discursos en distintos formatos, con la aplicación de técnicas de ETL y de procesamiento de lenguaje natural, que incluyó tokenización, lematización, etiquetado de partes de discurso (part-of-speech tagging) y reconocimiento de entidades (named entity recognition).

El uso de lenguaje R y del ecosistema de librerías Tidyverse, junto con RMarkdown, el módulo Spacy, una librería de código abierto para NLP avanzado que soporta más de 64 lenguajes humanos, nos han permitido desarrollar una prueba de concepto para poder implementar un análisis como prueba conceptual de la importancia de este tipo de análisis.

A través del análisis de los resultados, se ha comprobado que la utilización de las técnicas de análisis del lenguaje natural puede tener un gran potencia para el estudio de discursos, especialmente para aquellos investigadores que se acercan por primera vez a un período histórico, un movimiento político o un líder político, ya que permite identificar conceptos clave que pueden servir de guía para realizar una lectura más detenida o profunda con otras técnicas de análisis de discurso.

Como parte del análisis, se han podido detectar los principales temas del universo conceptual de Perón y el peronismo en el período analizado. Además, se pudieron detectar tanto regularidades como datos atípicos respecto de la extensión y periodicidad de los discursos.

Cabe señalar que lo presentado en este trabajo es escalable a todo tipo de documentación y tamaño del corpus que se desee analizar.

Un posible trabajo a futuro será buscar correlaciones entre la aparición de determinados tópicos en los discursos y otros factores de contexto histórico y social. En

el caso del período analizado, el conflicto con la Iglesia Católica, los problemas económicos por la sequía de 1951-1952 o los intentos de golpe de Estado.

También sería interesante sumar a estos análisis estadísticos, la participación de sociólogos o antropólogos, para enriquecer los resultados.

Finalmente, un aporte de este trabajo es también la puesta a disposición de la comunidad académica del corpus utilizado, luego del extenso trabajo de limpieza que se realizó previo al análisis. Dicho corpus puede ser accedido en https://github.com/martinolmos/discursos_peron.

# 9. Referencias Bibliográficas

Goel, B. (2017). Developments in The Field of Natural Language Processing. *International Journal of Advanced Research in Computer Science*, *8*(3), 23–28. Retrieved from www.ijarcs.info

Grimmer, J., & Stewart, B. M. (2013). Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts. *Political Analysis*, *21*(3), 267–297. https://doi.org/10.1093/pan/mps028

Perón, J. D. (2016). *Discursos, mensajes, correspondencia y escritos: 1949 / Perón* (Tomo I). Buenos Aires, Argentina: Biblioteca del Congreso de la Nación.

Turing, A. M. (2009). Computing Machinery and Intelligence. In R. Epstein, G. Roberts, & G. Beber (Eds.), *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer* (pp. 23–65). https://doi.org/10.1007/978-1-4020-6710-5_3

# 10. Anexos

## 10.1 Información de la Sesión

```
> sessionInfo()
R version 3.6.0 (2019-04-26)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19041)

Matrix products: default
locale:
[1] LC_COLLATE=Spanish_Argentina.1252  LC_CTYPE=Spanish_Argentina.1252
[3] LC_MONETARY=Spanish_Argentina.1252 LC_NUMERIC=C
[5] LC_TIME=Spanish_Argentina.1252

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] magrittr_2.0.1       tidytext_0.2.0       wordcloud_2.6
 [4] RColorBrewer_1.1-2   tm_0.7-6             NLP_0.2-0
 [7] lubridate_1.7.4      forcats_0.4.0        stringr_1.4.0
[10] dplyr_0.8.3          purrr_0.3.2          readr_1.3.1
[13] tidyr_0.8.3          tibble_2.1.1         ggplot2_3.3.2
[16] tidyverse_1.2.1.9000

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.6         cellranger_1.1.0  pillar_1.3.1      compiler_3.6
.0
 [5] tokenizers_0.2.1   tools_3.6.0       jsonlite_1.7.2    nlme_3.1-139
 [9] gtable_0.3.0       lattice_0.20-38   pkgconfig_2.0.2   rlang_0.4.10
[13] Matrix_1.2-17      cli_1.1.0         rstudioapi_0.10   parallel_3.6
.0
[17] haven_2.3.1        xfun_0.22         janeaustenr_0.1.5 withr_2.1.2
[21] xml2_1.2.0         httr_1.4.0        knitr_1.31        generics_0.0
.2
[25] vctrs_0.3.2        hms_0.4.2         grid_3.6.0        tidyselect_0
.2.5
[29] glue_1.4.2         R6_2.5.0          readxl_1.3.1      modelr_0.1.4
[33] SnowballC_0.6.0    backports_1.1.4   scales_1.0.0      rvest_0.3.3
[37] assertthat_0.2.1   colorspace_1.4-1  stringi_1.5.3     munsell_0.5.
0
[41] slam_0.1-45        broom_0.5.2       crayon_1.3.4
```

## 10.2. Código del Preprocesamiento y ETL

- Preprocesamiento de los archivos ".doc" (diferentes a ".DOC") Rich Text Format).

Levantamos los archivos:

```
library(tidyverse)
library(magrittr)
peron_doc <- readtext::readtext(
  file = "../../Discursos Presidentes/Peron/*.doc",
  encoding = "latin1")

Encoding(peron_doc$text) <- "latin1"
peron_doc2 <- peron_doc
```

Extraemos la fecha:

```
library(lubridate)
peron_doc2 %<>% mutate(fecha = str_extract(doc_id, "[[:digit:]]{6}"))
   %>%
     mutate(fecha = paste0("19", fecha)) %>%
     mutate(fecha = lubridate::as_date(x = fecha,
                                       format = "%Y%m%d",
                                       tz =
   "America/Argentina/Buenos_Aires"))
```

Eliminamos la primera línea que tiene la fecha:

```
peron_doc2 %<>% mutate(
    text = str_remove(text,
                      "^(\\r\\n)?( *)?[0-9].*\\r\\n"))
```

Eliminamos la primera línea en caso de que haya una línea vacía:

```
peron2 %<>% mutate(text = str_remove(text, "^\\r\\n"))
```

Extraemos el título:

```
peron_doc2 %<>% mutate(titulo = str_extract(text,
   "^.*(\\r\\n.*)?(\\r\\n.*)?\\r\\n(\\r\\n|  )"))
```

Sacamos los fin-de-línea que están al final de cada título y reemplazamos alguno que haya quedado en el medio por un espacio:

```
peron_doc2$titulo %>% str_extract("(?<=\\r\\n).*(?=:)")
```

Usamos regex para detectar las líneas que se colaron detrás del título:

```
peron_doc2 %<>% mutate(titulo = str_remove(titulo,
   "(?<=\\r\\n).*(?=:)"))
```

Eliminamos todos los fin-de-línea del título y los reemplazamos por un espacio. Luego "trimeamos" el título:

```
peron_doc2 %<>% mutate(titulo = str_replace_all(titulo, "\\r\\n", "
   ")) %>%
     mutate(titulo = str_trim(titulo, "both"))
```

Reemplazamos los fin-de-línea por espacios simples y luego reincorporamos los punto y aparte:

```
peron_doc2 %<>% mutate(text = str_replace_all(text, "\\r\\n", " "))
   %>%
     mutate(text = str_replace_all(text, "[.]     ", ".\r\n"))
```

- Preprocesamiento de los archivos ".DOC" (Rich Text Format).

```
myfiles <- list.files(path = "../../Discursos presidentes/Peron",
                      pattern = "*.DOC")
peron_DOC <- list()
tmp_names <- list()

for (i in seq_along(myfiles)) {
  peron_DOC[[i]] <- tryCatch(
    readtext::readtext(
      file = paste0("../../Discursos presidentes/Peron/", myfiles[i]),
      encoding = "latin1"
    ),
    error = function(cond) {
      message(paste("Error leyendo con readtext el archivo", myfiles[i
]))
      message(cond)
```

```
      message("Probando con textreadr")
      try({
        peron_DOC[[i]] <- textreadr::read_rtf(
          file = paste0("../../Discursos presidentes/Peron/", myfiles[
i]))
      })
    }
  )
}

names(peron_DOC) <- myfiles
```

Usamos readtext para levantarlos:

```
peron_DOCa <- peron_DOC[map(peron_DOC, function(x) class(x)[1]) == "re
adtext"]
peron_DOCb <- peron_DOC[map(peron_DOC, function(x) class(x)[1]) != "re
adtext"]

peron_DOCa <- bind_rows(peron_DOCa)

peron_DOCb <- peron_DOCb %>%
  map_df(~tibble(
    doc_id = "",
    text = paste(., collapse = "\n")
  )) %>%
  mutate(doc_id = names(peron_DOCb))

peron_DOC2 <- bind_rows(peron_DOCa, peron_DOCb)
```

- Preprocesamiento de los archivos ".rtf"

```
rtf_files <- list.files("../../Discursos presidentes/Peron/", "*.rtf")

peron_rtf <- rtf_files %>%
  map(function(x) textreadr::read_rtf(
          file = paste0("../../Discursos presidentes/Peron/", x)))

names(peron_rtf) <- rtf_files

peron_rtf <- peron_rtf %>%
  map_df(~tibble(
    doc_id = "",
    text = paste(., collapse = "\n")
```

```
  )) %>%
  mutate(doc_id = names(peron_rtf))

peron_DOC_rtf <- bind_rows(peron_DOC2, peron_rtf)

Encoding(peron_DOC_rtf$text) <- "latin1"
```

Extraemos la fecha:

```
library(lubridate)

peron_get_date <- function(df) {
  df %>%
    mutate(fecha = str_extract(doc_id, "[[:digit:]]{6}")) %>%
    mutate(fecha = paste0("19", fecha)) %>%
    mutate(fecha = lubridate::as_date(x = fecha,
                                      format = "%Y%m%d",
                                      tz = "America/Argentina/Buenos_Air
es"))
}
peron_rtf2 <- peron_get_date(peron_rtf)
```

Eiminamos la primera línea con la fecha:

Los rtf tienen '\n' en lugar de '\r\n' para señalar el fin-de-línea.

```
peron_rtf2 %<>% mutate(
    text = str_remove(text, "^(\\n)?( *)?[0-9].*\\n"))
```

Eiminamos la primera línea en caso que haya una línea vacía:

```
peron_rtf2 %<>% mutate(text = str_remove(text, "^\\n"))
```

Extraemos los títulos y los guardamos en nueva variable:

```
peron_get_titulo <- function(df) {
    df %>%
      mutate(titulo = str_extract(text, "^.*\\n"),
             text = str_remove(text, "^.*\\n"))
  }
  peron_rtf2 <- peron_get_titulo(peron_rtf2)
```

El único arreglo adicional que observamos es eliminar unos símbolos chinos (薄) que aparecen al final de cada discurso:

```
peron_rtf2 < peron_rtf2 %>%
    mutate(text = str_remove(text, "薄+"))
```

- Preprocesamiento de los archivos ".pdf"

```
library(pdftools)

peron_pdf <- pdf_text("../../Discursos presidentes/Peron/publicacionPeron-1949tomo1.pdf")

peron_pdf_indice <- peron_pdf[6:10]
peron_pdf_discursos <- peron_pdf[40:445]
```

Del índice extraemos la fecha y el título:

```
peron_pdf_indice_df <- peron_pdf_indice %>%
  paste(collapse = "\\r\\n") %>%
  str_extract_all(".*[0-9]{2,3}\\r\\n(  )?\\(.*\\)") %>% unlist() %>%
  {tibble(titulo = str_extract(.,".*(?= [0-9]*\\r\\n)") %>% unlist(),
          pagina = str_extract(.,"[0-9]+(?=\\r\\n(  )?\\()") %>% unlist(),
          fecha = str_extract(.,"\\(.*\\)") %>% unlist())}

peron_pdf_indice_df %<>% mutate(
  titulo = str_remove_all(titulo, "\\\\r\\\\n"),
  pagina = str_remove_all(pagina, "\\r\\n"),
  fecha = str_remove_all(fecha, "\\r\\n")) %>%
  mutate(
    titulo = str_trim(titulo),
    pagina = str_trim(pagina),
    fecha = str_trim(fecha)
  ) %>%
  mutate(
    fecha = str_remove_all(fecha, "[()]")
  )
```

Agregamos una columna con la última página de cada capítulo y sumamos uno a la variable página por el desfasaje:

```
peron_pdf_indice_df %<>% mutate(ultima = 0)
for (i in 1:nrow(peron_pdf_indice_df)) {
```

```
  peron_pdf_indice_df$ultima[i] <- as.numeric(peron_pdf_indice_df$pagi
na[i + 1])
peron_pdf_indice_df$pagina[i] <- as.numeric(peron_pdf_indice_df$pagina
[i]) + 1
}
peron_pdf_indice_df$ultima[61] <- 443
```

Ahora trabajamos el texto aparte:

```
peron_pdf[40] %>%
  str_remove("\\r\\n( +)?[0-9]+\\r\\n +.*(\\r\\n.*)*(\\r\\n)?$")
```

Eliminamos los pie de página:

```
peron_pdf2 <- peron_pdf

peron_pdf2 <- peron_pdf %>%
  map_chr(~str_remove(.,"\\r\\n( +)?[0-9]+\\r\\n +.*(\\r\\n.*)*(\\r\\n
)?$"))
```

Eiminamos los tramos de los pie de página que se extendieron a la página siguiente:

```
peron_pdf2[41] <- peron_pdf2[41] %>%
  str_remove("\\r\\n.*participado, por encargo de Perón.*(\\r\\n.*)*\\
r\\n$")

peron_pdf2[85] <- peron_pdf2[85] %>%
  str_remove("\\r\\n(.*)?mental para los trabajadores del sector.*\\r\
\n.*\\r\\n")

peron_pdf2[159] <- peron_pdf2[159] %>%
  str_remove("\\r\\n(.*)?una operación de apéndice.*\\r\\n.*\\r\\n$")

peron_pdf2[168] <- peron_pdf2[168] %>%
  str_remove("\\r\\n(.*)?sentados distintos sectores.*(\\r\\n.*)*(\\r\
\n)?$")

peron_pdf2[195] <- peron_pdf2[195] %>%
  str_remove("\\r\\n(.*)?mo Perón, en una entrevista concedida.*(\\r\\
n.*)*(\\r\\n)?$")
```

Eliminamos los números de pagina:

```
peron_pdf2 <- peron_pdf2 %>%
  map2_chr(.x = .,
           .y = as.character(2:446),
```

```
            .f = function(x, y) {
              str_remove(x, y)
            })
```

Pegamos las páginas de un mismo discurso (cambiamos las variables página y última a numeric):

```
peron_pdf_indice_df %<>% mutate(pagina = as.numeric(pagina),
                                ultima = as.numeric(ultima))

peron_pdf2 <- map2_chr(peron_pdf_indice_df$pagina,
                       peron_pdf_indice_df$ultima,
                       function(x, y) {
                         paste(peron_pdf2[x:y],
                               collapse = "\r\n")
                       })
```

Eliminamos fecha y titulo del texto:

```
peron_pdf2 <- peron_pdf2 %>%
  map_chr(.,function(x) {
    str_remove(x, "^.*\\r\\n.*(\\r\\n.*)?[A-Z]\\w*.*[0-9]+\\r\\n") %>%
      str_trim(side = "both")
  }
```

Eliminamos las palabras cortadas, los fin que no corresponden y los espacios sin sentido:

```
peron_pdf2 <- peron_pdf2 %>%
  map_chr(., function(x) {
    x %>%
      str_remove_all("-\\r\\n( *)") %>%
      str_replace_all("\\r\\n", " ") %>%
      str_replace_all("\\. {4,}", ".\r\n    ") %>%
      str_replace_all(": {4,}", ":\r\n    ") %>%
      str_replace_all(" {4,}", " ")
  })
```

Agregamos el texto al dataframe y transformamos la fecha al formato correspondiente:

```
peron_pdf_df %<>%
  mutate(
    text = peron_pdf2,
    fecha = lubridate::dmy(paste(fecha, 1949))) %>%
  select(-texto)
```

```
peron_pdf_df <- peron_pdf_df %>%
  mutate(fecha = lubridate::dmy(paste(fecha, 1949))) %>%
  rename(text = texto)
```

Aplicamos todo el mismo proceso al segundo lote de archivos .pdf, con los discursos de Perón de 1949:

```
library(pdftools)

peronB_pdf <- pdf_text("../../Discursos presidentes/Peron/publicacionPeron-1949Tomo-II.pdf")

peronB_pdf_indice <- peronB_pdf[6:11]
peronB_pdf_discursos <- peron_pdf[12:425]

peronB_pdf_indice_df <- peronB_pdf_indice %>%
  paste(collapse = "\\r\\n") %>%
  str_extract_all(".*[0-9]{2,3}\\r\\n(  )?\\(.*\\)") %>% unlist() %>%
  {tibble(titulo = str_extract(.,".*(?= [0-9]*\\r\\n)") %>% unlist(),
          pagina = str_extract(.,"[0-9]+(?=\\r\\n(  )?\\()") %>% unlist(),
          fecha = str_extract(.,"\\(.*\\)") %>% unlist())}

peronB_pdf_indice_df %<>% mutate(
  titulo = str_remove_all(titulo, "\\\\r\\\\n"),
  pagina = str_remove_all(pagina, "\\r\\n"),
  fecha = str_remove_all(fecha, "\\r\\n")) %>%
  mutate(
    titulo = str_trim(titulo),
    pagina = str_trim(pagina),
    fecha = str_trim(fecha)
  ) %>%
  mutate(
    fecha = str_remove_all(fecha, "[()]")
  )
```

```
peronB_pdf_indice_df %<>% mutate(ultima = 0)

for (i in 1:nrow(peronB_pdf_indice_df)) {
  peronB_pdf_indice_df$ultima[i] <- as.numeric(peronB_pdf_indice_df$pagina[i + 1])
  peronB_pdf_indice_df$pagina[i] <- as.numeric(peronB_pdf_indice_df$pagina[i]) + 1
}
peronB_pdf_indice_df$ultima[64] <- 416

peronB_pdf2 <- peronB_pdf
```

```
peronB_pdf2 <- peronB_pdf %>%
  map_chr(~str_remove(.,"\\r\\n( +)?[0-9]+\\r\\n +.*(\\r\\n.*)*(\\r\\n
)?$"))

peronB_pdf2[139] <- peronB_pdf2[139] %>%
  str_remove("\\r\\n.*para el gobierno presidido.*\\r\\n.*$")

peronB_pdf2[249] <- peronB_pdf2[249] %>%
  str_remove("\\r\\n.*el proyecto de un monumento.*\\r\\n.*$")

peronB_pdf2[340] <- ""

peronB_pdf2[341] <- peronB_pdf2[341] %>%
  str_remove("\\r\\n.*certifica asimismo.*(\\r\\n.*)*$")



peronB_pdf2 <- peronB_pdf2 %>%
  map2_chr(.x = .,
           .y = as.character(2:426),
           .f = function(x, y) {
             str_remove(x, y)
           })

peronB_pdf_indice_df %<>% mutate(pagina = as.numeric(pagina),
                                 ultima = as.numeric(ultima))

peronB_pdf2 <- map2_chr(peronB_pdf_indice_df$pagina,
                  peronB_pdf_indice_df$ultima,
                  function(x, y) {
                    paste(peronB_pdf2[x:y],
                          collapse = "\r\n")
                  })

peronB_pdf2 <- peronB_pdf2 %>%
  map_chr(.,function(x) {
    str_remove(x, "^.*\\r\\n.*(\\r\\n.*)?[A-Z]\\w*.*[0-9]+\\r\\n") %>%
      str_trim(side = "both")
  })

peronB_pdf2 <- peronB_pdf2 %>%
  map_chr(., function(x) {
    x %>%
      str_remove_all("-\\r\\n( *)") %>%
      str_replace_all("\\r\\n", " ") %>%
      str_replace_all("\\. {4,}", ".\r\n    ") %>%
      str_replace_all(": {4,}", ":\r\n    ") %>%
      str_replace_all(" {4,}", " ")
  })
```

```
peronB_pdf_df <- peronB_pdf_indice_df %>%
  mutate(
    text = peronB_pdf2,
    fecha = lubridate::dmy(paste(fecha, 1949)))
```

Finalmente, unificamos todos los dataframes obtenidos parcialmete en uno solo:

```
peron_dfs <- list(peron_doc2, peron_DOC_rtf, peron_pdf_df, peronB_pdf_
df)

peron_df <- peron_dfs %>%
  map(~select(., fecha, titulo, text)) %>%
  bind_rows()

write_csv(peron_df, "peron_discursos_completos.csv")
```

## 10.3 Procesamiento NPL

```
import pandas as pd
import spacy

peron_df = pd.read_csv('C:/Users/marti/Documents/Ciencia de Datos/Taller/peron_discursos_completos.csv', encoding = 'latin1')

spacy_es = spacy.load('es_core_news_sm')

def doc_to_df(doc, i):
    df = pd.DataFrame()
    df['text'] = [token.text for token in doc]
    df['lemma'] = [token.lemma_ for token in doc]
    df['is_punctuation'] = [token.is_punct for token in doc]
    df['is_space'] = [token.is_space for token in doc]
    df['shape'] = [token.shape_ for token in doc]
    df['part_of_speech'] = [token.pos_ for token in doc]
    df['pos_tag'] = [token.tag_ for token in doc]
    df['id'] = i
    return df

def ents_to_df(doc, i):
    df = pd.DataFrame()
    df['ent'] = [ent.text for ent in doc.ents]
    df['label'] = [ent.label_ for ent in doc.ents]
    df['id'] = i
    return df

for i, row in peron_df.iterrows():
    doc = spacy_es(row.text)
    doc_df = doc_to_df(doc, i)
    doc_df.to_csv('C:/Users/marti/Documents/Ciencia de Datos/Taller/annot/doc_%s.csv' % (str(i)), index=False)
    ents_df = ents_to_df(doc, i)
    ents_df.to_csv('C:/Users/marti/Documents/Ciencia de Datos/Taller/annot/ents_%s.csv' % (str(i)), index=False)
```

```
peron_annot_docs <- list()
peron_annot_ents <- list()

peron_annot_docs_files <- list.files("annot/", "doc.*\\.csv")
peron_annot_ents_files <- list.files("annot/", "ents.*\\.csv")

for (i in seq_along(peron_annot_docs_files)) {
  peron_annot_docs[[i]] <- read_csv(paste0("annot/", peron_annot_docs_files[[i]]))
}
```

```
peron_annot_docs_df <- bind_rows(peron_annot_docs)

for (i in seq_along(peron_annot_ents_files)) {
  peron_annot_ents[[i]] <- read_csv(paste0("annot/", peron_annot_ents_
files[[i]]))
}

peron_annot_ents_df <- bind_rows(peron_annot_ents)
```