Proyecto de Grado
Ingeniería en Computación

Procesamiento de Lenguaje Natural (PLN) para la reconstrucción de textos a partir de imágenes correspondientes a archivos históricos de la década del 70

Javier Stabile
javier.stabile@fing.edu.uy
Ernesto Fernandez
ernesto.fernandez@fing.edu.uy
Federico Fioritto
federico.fioritto@fing.edu.uy

Tutoras:
Aiala Rosá
Dina Wonsever

Facultad de Ingeniería
Universidad de la República
26 de noviembre de 2020

Resumen

Durante los años 70's y mediados de los 80's, en Uruguay tuvo lugar la dictadura cívico-militar a través de un golpe de estado. A raíz de esto, los poderes del estado pasan a manos de militares, se pierden derechos y libertades por parte de la población, y ocurren detenciones políticas y desaparición de personas.

Durante este periodo la documentación publica era muy limitada y no se conocían con claridad los hechos que ocurrían. Recientemente se liberaron numerosos documentos conservados en formato microfilm que han perdido calidad con los años transcurridos. Lo que genera dificultades para ser legibles al ser escaneados. El procesamiento de estos documentos es de suma importancia para la dilucidación de eventos ocurridos durante el periodo mencionado.

Este proyecto se realizó en el marco de un proyecto más amplio de recuperación de archivos de texto, en formato imagen de escasa legibilidad.

Se propone la aplicación de técnicas de Procesamiento de Lenguaje Natural (PLN) para la reconstrucción de textos a partir de imágenes correspondientes a archivos históricos de las décadas mencionadas anteriormente. Se cuenta con una importante base de imágenes escaneadas, algunos procesamientos realizados sobre estas imágenes y la transcripción manual del texto contenido en diferentes segmentos de un subconjunto de imágenes (proyecto LUISA).

Se busca alcanzar diferentes objetivos como son mejorar los resultados de la herramienta que se usa para convertir imagen a texto (OCR), reconstruir como texto documentos enteros a partir de segmentos transcritos en forma manual, que se están generando mediante un proceso de anotación colaborativa, y por último, corregir mediante técnicas de PLN las extracciones de los textos escaneados y procesados por un OCR.

Índice general

Capítulo 1

Introducción

En el año 1973 en Uruguay ocurre un golpe de estado por parte de las fuerzas armadas. Durante los siguientes años hasta 1985, se tuvo una dictadura cívico-militar, con la que se llegó a la militarización de los poderes del estado, a la paulatina pérdida de las libertades de la población, a detenciones políticas y a la desaparición de personas.

Al estar los poderes del estado en manos militares no se tenía documentación pública sobre los hechos ocurridos en el país hasta la desclasificación reciente de numerosos documentos. Estos documentos son de suma importancia para esclarecer muchos de los hechos ocurridos en esta oscura etapa de nuestro país. Los mismos fueron conservados en formato de microfilm y con el paso del tiempo se han ido deteriorando y perdiendo calidad, lo que dificulta su lectura al ser escaneados. Sumado a esto, el escaneo se hizo rápidamente y en binario, perdiéndose de esta forma detalles como la escala de grises. Por otro lado la cantidad de documentos es muy grande, lo que conllevaría un gran esfuerzo para su procesamiento en caso de que se hiciese de forma manual, esto incentiva la idea de automatizarlo. El procesamiento de estos documentos no solo es importante para conocer mejor la historia de nuestro país, sino que para dar algunas respuestas a las familias detrás de las personas desaparecidas.

En pro de lo antes mencionado, se desarrolló el proyecto LUISA (Leyendo Unidos para Interpretar loS Archivos), el cual tiene como finalidad ayudar al procesamiento de documentos escaneados ofreciendo una plataforma colaborativa *online*, en la cual la comunidad transcribe individualmente pequeños bloques o segmentos de documentos del tamaño de palabras, los cuales al reagruparse terminan construyendo representaciones de los documentos enteros. De esta forma, al dividir en partes pequeñas las imágenes, ningún individuo tiene acceso a un documento entero y se preserva su confidencialidad.

Para la realización de nuestro proyecto se cuenta con una base muy importante de imágenes de documentos ya escaneados y sus transcripciones manuales realizadas por la comunidad mediante el proyecto LUISA.

Nuestro proyecto tiene como finalidad aportar al procesamiento automático de los documentos mejorando la calidad de los resultados obtenidos.

Objetivo General

Es así que el objetivo general de este proyecto se define como la investigación de técnicas pertenecientes al campo del procesamiento del lenguaje natural para la corrección de textos históricos que vieron su origen durante la dictadura militar ocurrida en nuestro país en las décadas de 1970 y 1980.

Objetivos Específicos

A continuación presentamos una lista de los objetivos específicos que se persiguieron en este proyecto con el fin de cumplir el objetivo general. Algunos de ellos fueron planteados desde las etapas iniciales de la investigación, mientras que otros surgieron según los requerimientos propios de las decisiones tomadas durante el desarrollo de esta.

- Comparación de distintos sistemas de OCR, incluyendo al seleccionado inicialmente en la propuesta del proyecto con el fin de generar textos procesables con la mejor calidad posible.
- Entrenamiento del sistema de OCR *opensource* seleccionado en la propuesta del proyecto.
- Reconstrucción de documentos de texto a partir de transcripciones aportadas por la plataforma LUISA de recortes de bloques aislados de los documentos digitales. Este nuevo conjunto de datos conformará el *Ground Truth* para futuras etapas.
- Implementación y evaluación de técnicas de corrección del texto generado mediante sistemas de OCR.
- Comparación del rendimiento de las diferentes estrategias y variantes investigadas para la corrección de textos durante el proyecto.

Esquema general del proceso de corrección

En la imagen 1.1 se muestra un esquema en alto nivel que describe el proceso completo de corrección. Se parte desde los documentos digitalizados como señala el indicador en rojo. Estos pasan a ser procesados por el OCR utilizado en este proyecto además de alimentar el proyecto LUISA (sistema externo a esta investigación). El OCR genera textos procesables los cuales son corregidos por las distintas técnicas de corrección implementadas. Finalmente, las correcciones son evaluadas por un último módulo de evaluación. Este hace uso de un *Ground Truth* construido a partir de las transcripciones aportadas por el proyecto LUISA.

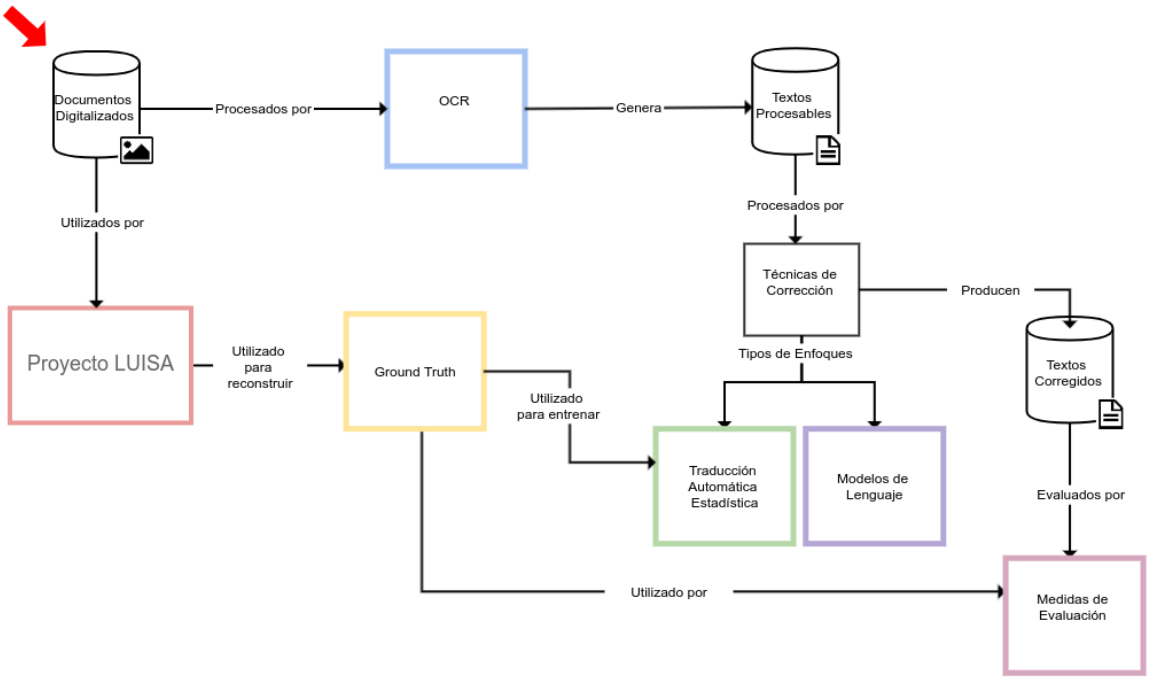


Imagen 1.1: Esquema general del proceso

Para ver un esquema más detallado del procesamiento completo de los documentos referirse al diagrama 10.1 adjunto en el anexo.

Estructura del informe

El resto del informe tiene la siguiente estructura:

El capítulo dos presenta un abordaje teórico sobre el reconocimiento óptico de caracteres, modelos de lenguaje y traducción automática estadística. También se presentan las medidas *ratio* y *BLEU* utilizadas para evaluar los resultados y algunos trabajos previos sobre la temática abordada.

En el tercer capítulo se presenta el *Ground truth* utilizado para validar, la procedencia de los datos con los que se cuenta, su construcción, las dificultades enfrentadas al construirlo y su evaluación.

En el capítulo cuatro se presentan los distintos OCRs candidatos utilizados para escanear los documentos históricos, al final del capítulo se realiza una comparativa y se decide el más adecuado.

El capítulo cinco presenta una de las soluciones planteadas para el problema de corregir las salidas obtenidas por el OCR, esta solución es la basada en modelos de lenguaje. Se detallan los distintos modelos utilizados, el procesamiento realizado y las decisiones tomadas.

El capítulo seis introduce una solución alternativa basada en traducción automática estadística detallando desde la creación de un corpus paralelo, la alineación de textos, el funcionamiento de la traducción y entrenamiento necesario.

En el capítulo siete se presentan los resultados utilizando las dos soluciones, modelo de lenguaje y traducción automática estadística.

El capítulo ocho contiene las conclusiones logradas referente al trabajo.

En el capítulo nueve se presenta una visión de los trabajos a futuro.

Capítulo 2

Marco teórico

En el presente proyecto, se hace uso de múltiples conceptos, estudios e investigaciones realizadas a lo largo del tiempo. En este capítulo se presenta y explica el respaldo teórico utilizado.

2.1. Reconocimiento óptico de caracteres

El proceso completo que estaremos describiendo comienza tomando como entrada imágenes digitales, sin embargo, las distintas técnicas de corrección implementadas trabajan con texto entendible por una computadora, entonces resulta natural la pregunta ¿cómo se obtiene el texto procesable a partir de las imágenes digitales? La respuesta es: se logra a través de la aplicación de un proceso llamado reconocimiento óptico de caracteres, comúnmente conocido por su nombre en inglés *optical character recognition* o por sus siglas OCR.

Entonces, el reconocimiento de caracteres puede definirse como el proceso de conversión de texto escaneado, imágenes impresas o escritura a mano, en texto editable para futuros procesamientos [1]. Este no es un problema para nada nuevo, las primeras automatizaciones de estos procesos datan de la década de 1950 y desde entonces se han estado desarrollando y mejorando las técnicas que lo resuelven, a pesar de ello, estos sistemas se encuentran muy lejos de competir con el ojo humano.

Aplicación

En los últimos años, el software de OCR ha sido aplicado en una gran variedad de industrias, como puede ser el ámbito de entidades legales, donde se ha visto una tendencia a digitalizar los documentos físicos con el objetivo de ahorrar espacio y tiempo, eliminando la necesidad de búsquedas entre montañas de papeles. De forma similar, la industria de la salud ha mostrado estar cada vez más interesada en la aplicación de estas técnicas para evitar el uso de formularios físicos, agilizando consultas de la información, ya que la misma se encontrará accesible en bases de datos por medios electrónicos. Su utilización también es conocida en el área bancaria, dónde permite el procesamiento de cheques sin que haya humanos involucrados. Y podríamos nombrar un gran número de casos más dentro de los campos de la educación, finanzas, seguridad de sistemas de software, librerías digitales, etc. [2]

Tipos de OCR

El amplio dominio de aplicación de los reconocedores ha permitido el surgimiento de diferentes tipos de sistemas. Estos se pueden categorizar según varias características como: restricciones de las fuentes

encontradas en los documentos a procesarse, métodos de adquisición de imágenes, tipos de datos de entrada, etc. [3]

Basándose en el tipo de entrada (Imagen 2.1), encontramos los reconocedores de caracteres impresos a máquina (machine-printed OCR) y los reconocedores de caracteres escritos a mano (handwritten OCR). Los primeros atacan un problema virtualmente más sencillo ya que la disposición y forma de los caracteres suele ser uniforme, lo que facilita el descubrimiento de patrones y la predicción de posiciones.

Los OCR que trabajan con escritura a mano se encuentran en una situación más complicada debido a los diferentes estilos de escritura. Estos sistemas pueden ser divididos en dos sub-categorías: online y offline. El primero es ejecutado en tiempo real mientras que los usuarios están escribiendo los caracteres. El reconocimiento offline por otro lado opera con datos estáticos como pueden ser cartas o cheques.

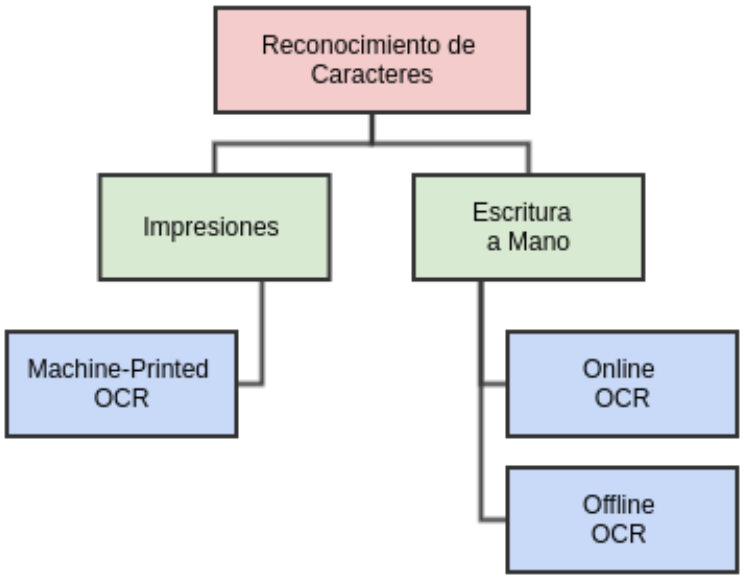


Imagen 2.1: Tipos de reconocedores ópticos de caracteres

Pasos principales en el reconocimiento

La mayoría de los sistemas de reconocimiento óptico de caracteres siguen una estructura de procesamiento similar. A continuación, finalizamos esta breve introducción al reconocimiento óptico de caracteres listando los principales pasos en común presentes en este tipo de procesos.

1. Obtención de la imagen.
2. Preprocesamiento: Una vez que se cuenta con la imagen, distintos preprocesamientos pueden ser ejecutados para mejorar la calidad de la imagen y facilitar la extracción de la información. Una de las técnicas más utilizadas es la binarización la cual consiste en transformar imágenes que estén a color o en escalas de grises a imágenes en blanco y negro, delimitando así de forma más clara los contornos de los caracteres.
3. Segmentación: Se aplica un enfoque top-down, primero se reconocen las líneas del documento, luego se extraen las palabras y finalmente cada palabra es segmentada en caracteres.

4. Extracción de Características: Los caracteres segmentados obtenidos en el paso anterior son procesados para extraer diferentes características a partir de las cuales el carácter puede ser reconocido. Un ejemplo es la extracción de características basada en zonas. En la misma la imagen es dividida en zonas tanto vertical como horizontalmente, y para cada zona resultante se calcula la densidad de píxeles que el carácter posee. En la Imagen 2.2 se puede ver del lado izquierdo, la imagen de un carácter resultado de la segmentación y a la derecha vemos un análisis basado en zonas del mismo carácter, dónde los cuadrados más oscuros significan mayor densidad de píxeles [4].

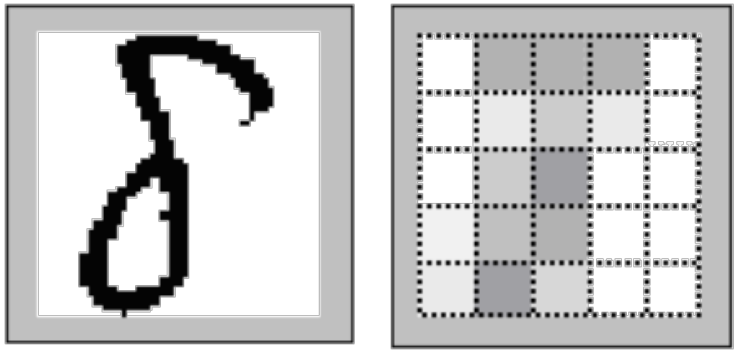


Imagen 2.2: Extracción de características basada en zonas

5. Clasificación: En este paso se clasifican los caracteres en su categoría apropiada basándose en el análisis de las características extraídas en el paso anterior.
6. Postprocesamiento: Finalmente, una vez que un carácter ha sido clasificado, se pueden utilizar diversos enfoques para mejorar la precisión de los resultados del OCR. Una posible estrategia consiste en aplicar varios clasificadores en lugar de trabajar con un único clasificador. Los clasificadores pueden ser utilizados en cascada o paralelo. Luego, los resultados de cada clasificación pueden ser combinados.

2.2. Modelos de lenguaje

Los modelos de lenguaje son modelos estadísticos que tienen la finalidad de determinar por medio de probabilidades la probabilidad de una secuencia de palabras o la siguiente palabra a partir de una secuencia de palabras. Son muy utilizados en aplicaciones computacionales como el *speech recognition* o reconocimiento del habla, traducción automática, reconocimiento de escritura o recuperación de la información entre otras aplicaciones. Los modelos más básicos trabajan a nivel de unas pocas palabras, mientras que los más grandes funcionan a nivel de oraciones o párrafos. La mayoría operan a nivel de palabras.

Dada una secuencia de palabras de largo m, un modelo de lenguaje asigna una probabilidad $P(w_1,\ldots,w_m)$ a la secuencia de palabras. La manera de calcular la probabilidad P es distinta según el tipo de modelo de lenguaje que se tenga.

Un modelo de lenguaje aprende la probabilidad de una palabra o frase basado en los datos de entrenamiento: cuanto más datos, mayor será su precisión. Los datos son interpretados a través de algoritmos específicos según el propósito del modelo de lenguaje. Los modelos de lenguaje diseñados para predecir palabras, detectar mails de *spam* o automatizar respuestas son entrenados de manera distinta, debido a que son distintos los patrones del lenguaje que se quieren identificar en los datos.

En el caso *speech recognition*, donde varias palabras son fonéticamente similares, por medio de modelos de lenguaje se logra determinar la opción más correcta utilizando el contexto. Teniendo como ejemplo la secuencia de palabras "El continente" y las candidatas a tercera palabra "Asia" y "hacia", un modelo entrenado adecuadamente debería determinar que

$$P(El, continente, Asia) > P(El, continente, hacia)$$

por lo tanto elegir la opción "Asia". Un ejemplo de predicción con modelos de lenguaje son los teclados de los Smartphones, que dada una secuencia de palabras sugiere la siguiente, calculando la probabilidad del contexto junto con todas las palabras candidatas a seguir. La secuencia con mayor probabilidad es la sugerida.

Existen varios tipos de modelo de lenguaje, entre los más conocidos están unigrama, n-grama, exponencial y de redes neuronales.

A continuación se describirán los relacionados con este proyecto.

N-grama

A diferencia de un modelo de unigrama, un modelo de lenguaje de n-gramas asigna probabilidades a una secuencia de palabras. El parámetro n puede ser cualquier entero positivo y define el largo de secuencias a considerar.

De acuerdo al capítulo tres del libro "Speech and Language Processing" [5], para el modelo de n-gramas la probabilidad P(w_1,...,w_m) de la frase w_1,...,w_m se calcula como:

$$P(w_1, \ldots, w_m) = \prod_{i=1}^{m} P(w_i | w_1, \ldots, w_{i-1}) \approx \prod_{i=1}^{m} P(w_i | w_{i-(n-1)}, \ldots, w_{i-1})$$

Por el orden *n-ésimo* de la propiedad de Markov, se asume que la probabilidad de observar la *i-ésima* palabra w_i en el contexto de las palabras i-1 anteriores se aproxima a la probabilidad de observar el contexto de las n-1 anteriores palabras. La probabilidad condicional puede calcularse a partir de conteos de frecuencia n-grama:

$$\prod_{i=1}^{m} P(w_i | w_{i-(n-1)}, \ldots, w_{i-1}) = \frac{count(w_{i-(n-1)}, \ldots, w_{i-1}, w_i)}{count(w_{i-(n-1)}, \ldots, w_{i-1})}$$

Para evitar probabilidades nulas al tener un n-grama no visto antes, se deben aplicar técnicas de suavizado, como agregar una observación a todos los elementos del vocabulario o utilizar modelos como *Good-Turning*, *back-off* o interpolación.

Para un modelo de bigrama o de n=2, la probabilidad de la frase "Una gran frase escrita" se determina como:

$$P(Una, gran, frase, escrita) \approx P(Una| < s >)P(gran|Una)P(frase|gran)P(escrita|frase)P(< /s > |escrita)$$

Para un modelo de trigrama o n = 3, la misma frase se aproxima como:

$$P(Una, gran, frase, escrita) \approx$$
$$P(Una| < s >, < s >)P(gran| < s >, Una)P(frase|Una, gran)P(escrita|gran, frase)P((< /s > |frase, escrita)$$

Los símbolos $< s >$ y $< /s >$ se utilizan como marcadores de inicio y final de frase.

Redes neuronales

Los modelos de lenguaje neuronales o de espacio continuo usan representaciones continuas o *embeddings* de palabras para hacer sus predicciones, y lo hacen utilizando redes neuronales. La utilización de *embeddings* es la representación de palabras como vectores de números reales.

Usar *embeddings* es muy útil para conjuntos de datos enormes, dado que estos usualmente presentan mayor cantidad de palabras únicas o raras. El número de secuencias de palabras posibles aumenta exponencialmente con el tamaño del conjunto de datos, esto genera dispersión de datos.

La presencia de palabras únicas causa problemas para los modelos lineales como el n-gramas. Esto debido a que la cantidad de secuencias de palabras comunes son muy grandes, pero no para las secuencias con palabras únicas, lo que genera un pobre resultado cuando ocurre una palabra poco común. Representando las palabras de una forma distribuida y no lineal, el modelo es capaz de aproximar palabras y por lo tanto ser robusto ante palabras únicas o desconocidas. La predicción de una palabra dada no está tan estrechamente ligada a las palabras circundantes como lo están en los modelos de n-gramas.

Generalmente los modelos neuronales son construidos y entrenados como clasificadores probabilísticos que aprenden a predecir una distribución probabilística. Son redes neuronales *standard* entrenadas a través de algoritmos de descenso por gradiente con *backpropagation*. Los más conocidos son *bag-of-word* y *skip-gram* [6] [7].

2.3. Traducción automática estadística

La traducción automática estadística [8] (comúnmente conocida por las siglas SMT del inglés *Statistical Machine Translation*) es uno de los paradigmas que encontramos dentro del área de la lingüística computacional denominada **traducción automática**.

Los inicios de la traducción automática datan tan solo unos pocos años después de que las primeras computadoras fueran utilizadas para resolver problemas de cifrado durante la segunda guerra mundial, partiendo de la analogía de que la tarea de descifrar un mensaje a su idioma original no distaba tanto de la de traducir un texto desde un idioma S a otro idioma T. El campo floreció y mantuvo un nivel de popularidad alto hasta que en 1966 un reporte de la ALPAC (*Automatic Language Processing Advisory Committee*) que veía con ojos negativos a la traducción automática provocó la detención del desarrollo y el abandono de las investigaciones en esta área durante 20 años. No es hasta finales de la década de los 80 que la traducción automática experimenta un resurgimiento gracias a los avances y resultados mostrados por investigadores de IBM. A partir de estos años, el enfoque estadístico de la traducción automática se vuelve la estrategia dominante en el área por varias décadas, hasta el surgimiento de la traducción automática neuronal, la cual ha ganado popularidad en los últimos años.

Las traducciones generadas por los sistemas de SMT se basan en modelos estadísticos construidos a través del análisis de corpus bilingües, también conocidos como corpus paralelos. Estos corpus son conjuntos de textos en dos idiomas diferentes que se encuentran alineados entre sí.

Es usual recurrir a la utilización de probabilidades cuando se intenta resolver problemas relacionados a eventos con cierto nivel de incertidumbre, como el caso de una palabra en un idioma S que puede ser traducida a varias palabras distintas en un idioma T. Estos eventos pueden ser modelados bajo distintas distribuciones probabilísticas. Las distribuciones probabilísticas no son más que funciones que asignan un valor entre 0 y 1 a las posibles salidas que toma el evento. Bajo el contexto de la

traducción automática, el problema de seleccionar qué distribución emplear para modelar los eventos se resuelve por medio del análisis de los corpus con la aplicación del método de estimación por máxima verosimilitud.

Luego, un documento es traducido de acuerdo a la distribución de probabilidad $P(t|s)$ de que un texto t en el idioma destino sea la traducción del texto s en el idioma origen. Es común que las distribuciones de probabilidad sean reformuladas con el teorema de Bayes el cual expresa la probabilidad condicional de un evento aleatorio A dado B en términos de la distribución de probabilidad condicional del evento B dado A y la distribución de probabilidad marginal de solo A. De esta manera, luego de aplicar Bayes sobre la distribución original se obtiene:

$$P(t|s) = \frac{P(s|t) \cdot P(t)}{P(s)} \tag{2.1}$$

Esta nueva fórmula (2.1) nos dice que la probabilidad de que un texto t del idioma destino sea la traducción de s es igual a la probabilidad de que s sea la traducción de t multiplicado por la probabilidad de encontrarnos a t en el idioma destino, dividido por la probabilidad de encontrar al texto s en el idioma origen. Aquí es importante mencionar dos detalles: en primer lugar, notar que las probabilidades $P(s)$ y $P(t)$ indican qué tan verosímil es que esos textos estén presentes en sus respectivos lenguajes, por lo tanto refieren a probabilidades dadas por modelos de lenguaje. En segundo lugar, se tiene que s es el texto a traducir, el cual es una constante entre todas las posibles traducciones t, por lo que a la hora de comparar qué traducción t es más verosímil, podemos eliminar a $P(s)$ de la ecuación. Generalizando obtenemos que para un texto s perteneciente al idioma origen S y un conjunto de posibles traducciones t pertenecientes al idioma destino T, la traducción final $\bar{t}$ será seleccionada según la siguiente fórmula:

$$\bar{t} = \underset{t \in T}{\operatorname{argmax}} P(t|s) = \underset{t \in T}{\operatorname{argmax}} P(s|t) \cdot P(t) \tag{2.2}$$

En el artículo *The Mathematics of Statistical Machine Translation: Parameter Estimation* [9] se hace referencia a la fórmula 2.2 como la ecuación fundamental de la traducción automática. También se menciona allí que presenta tres desafíos computacionales:

1. Estimar la probabilidad del modelo de lenguaje $P(t)$
2. Estimar la probabilidad del modelo de traducción $P(s|t)$
3. Definir una forma eficiente y efectiva de buscar entre las traducciones que maximicen el producto

Para resolver el desafío **1)** es necesario construir un modelo de lenguaje a partir del idioma destino. Este se encargará de ponderar el nivel de corrección y fluidez de las traducciones. Frases incorrectas tendrán una probabilidad $P(t)$ baja, provocando que su ponderación final también sea baja y descartándolas (idealmente) como posibles traducciones.

De manera similar a como ocurrió con el modelo de lenguaje, hallar una solución al desafío **2)** implica la construcción de un modelo, aunque en este caso es un modelo de traducción. Este modelo no tendrá como objetivo la generación de frases gramaticalmente correctas, eso será trabajo del modelo de lenguaje. El modelo de traducción utiliza estadísticas calculadas a partir de los corpus paralelos para determinar la probabilidad de que ciertas frases y palabras del idioma destino sean traducción de frases y palabras en el idioma origen.

Finalmente, resta solucionar el desafío número **3)**. Se necesita establecer una forma de búsqueda de traducciones que sea computacionalmente posible y pueda ser llevada a cabo en un tiempo razonable según el contexto del problema dentro del enorme conjunto de posibilidades. Encontrar la traducción dentro del idioma destino que maximice la probabilidad dada en 2.2 se llama decodificación (o *decoding* en inglés) y al módulo que implementa la búsqueda se lo conoce como decodificador (o *decoder*). Existe una gran variedad de algoritmos que puede ser utilizados para encontrar la mejor traducción dentro de las opciones. En el capítulo 6 veremos de forma general al algoritmo *Beam Search*.

A modo de cierre de la presente sección, resta mencionar que los sistemas SMT pueden ser empleados en un gran número de problemas, siendo uno de ellos la corrección de texto erróneo dónde se toma como lenguaje de origen el texto escrito incorrectamente y como lenguaje destino el texto escrito de manera correcta. Esta utilidad ciertamente novedosa es la que aplicamos en este proyecto buscando corregir la salida del OCR y en la cual se profundizará en el capítulo 6.

2.4. Medidas de evaluación

Medir el nivel de similitud entre textos en un proyecto enfocado a la corrección de documentos es un proceso de suma importancia ya que conforma el instrumento principal a la hora de evaluar resultados, contribuyendo a la comparación de enfoques o estrategias y agregando un valor fundamental al momento de tomar decisiones sobre qué rumbo seguir en cualquier investigación. En este proyecto comparamos los resultados obtenidos por nuestros diferentes procesamientos contra los resultados de las salidas sin procesar del OCR, en ambos casos midiendo la similitud con respecto a los datos que tomamos como *Ground Truth*.

Se emplearon dos algoritmos distintos de comparación de textos. Uno de ellos se enfoca en encontrar diferencias a nivel de caracteres y en la práctica es utilizado sobre cualquier pareja de textos, sean estos correcciones o traducciones de tamaño considerable, o simplemente líneas con pocas o tan solo una única palabra. El otro algoritmo que se manejó busca comparar a nivel n-gramas y trabaja con una medida estándar en el campo de las traducciones automáticas de textos. En las siguientes dos subsecciones los presentaremos, veremos las ideas generales que los caracterizan y detallaremos las herramientas que los implementan y fueron utilizadas en particular para este proyecto.

2.4.1. Similitud a nivel de caracteres

A la hora de comparar dos secuencias, la idea principal del algoritmo es la de buscar coincidencias o matcheos. Primero se intenta encontrar la subsecuencia de elementos **contiguos** más larga y luego aplicar esta misma idea recursivamente hacia la izquierda y derecha de la subsecuencia hasta haber recorrido el total de ambas entradas.

Esta forma de búsqueda de coincidencias guarda relación con el algoritmo *Longest Common Subsequence* o LCS con la diferencia de que el aplicado en este proyecto exige que la secuencia sea contigua, lo que genera matcheos "más naturales" para el ojo humano. Llamemos a esta variante de LCS como LCS_2. Veamos un ejemplo de los resultados producidos por ambos algoritmos para que quede más claro, consideremos las secuencias de entrada *"mi estereotipo misterioso"* y *"misterio la"*.

Aplicando LCS_2 obtenemos la siguiente coincidencia:

mi estereotipo **misterio**so
misterio la

Con LCS se tiene el siguiente resultado:

mi ester**e**ot**i**p**o** misterioso
misterio la

Para el caso de evaluar correcciones en nuestra investigación, consideramos más relevante las comparaciones del primer tipo (LCS_2).

Para pasar de la comparación de secuencias a números, se realiza el siguiente cálculo:

$$2.0 \cdot M/T$$

Donde T es el número total de caracteres en ambas secuencias y M es el número total de caracteres matcheados en cualquiera de las dos secuencias. Esto retorna una medida de la similitud entre las secuencias evaluadas como un número racional en el rango [0,1]. Por lo tanto, cuantos más caracteres coinciden, más alto es el valor de M y por consiguiente más cercano a 1 será el resultado final. De la misma forma, valores cercanos a 0 indicarán diferencias mayores entre las secuencias.

Pasando a hablar sobre herramientas en particular, la que implementa este algoritmo y fue utilizada en este proyecto es la clase SequenceMatcher [10] del módulo *difflib* de Python. La función que realiza el cálculo puntualmente luego de haber encontrado las subsecuencias en común lleva el nombre de *ratio()*. La documentación de la herramienta aclara como regla general que un valor de *ratio()* superior a 0.6 significa que los textos presentan un nivel de coincidencia alta.

Habiendo explicado la forma en que SequenceMatcher busca coincidencias, y cómo se traducen estas a números, nos parece importante ver algunos ejemplos para corroborar que el comportamiento es el buscado. De nuevo, las coincidencias encontradas por el algoritmo se remarcan en rojo.

Veamos primero comparaciones de errores ortográficos a nivel de palabra

secuencia 1	**integración**
secuencia 2	**integración**
ratio	1.0

secuencia 1	**integración**
secuencia 2	**integracion**
ratio	0.909

secuencia 1	**integración**
secuencia 2	**integrasion**
ratio	0.818

secuencia 1	**integración**
secuencia 2	**imtegrasion**
ratio	0.727

secuencia 1	**integración**
secuencia 2	**imtegrasiom**
ratio	0.636

secuencia 1	**integración**
secuencia 2	**imtégrasiom**
ratio	0.545

Como se puede apreciar, dos secuencias iguales tienen un ratio igual a 1.0, y a medida que ambas palabras se diferencian cada vez más a nivel de caracteres, el valor de ratio comienza a disminuir.

Debido a la forma en que el valor numérico de ratio es calculado, resulta interesante explorar el comportamiento sobre palabras más cortas. Las diferencias en estos casos deberían generar cambios más pronunciados en el resultado retornado.

secuencia 1	**días**
secuencia 2	**dias**
ratio	0.75

secuencia 1	**días**
secuencia 2	**diaz**
ratio	0.5

secuencia 1	**días**
secuencia 2	**día**
ratio	0.857

De esta manera verificamos el supuesto anterior. Esto tiene sentido ya que ratio básicamente calcula la proporción de caracteres distintos sobre el total de caracteres en ambas secuencias. Una diferencia

de X caracteres entre dos secuencias con largo total l, tendrá un mayor peso que una diferencia de también X caracteres sobre dos secuencias de largo total $L > l$.

Otro punto directamente relacionado a nuestro proyecto es el concerniente a la estructura de los textos. Entendiéndose como estructura a la cantidad de espacios en blanco, saltos de líneas, tabulaciones, sangrías y demás que dan forma a los documentos.

secuencia 1	**la luna**
secuencia 2	**la luna**
ratio	0.933

secuencia 1	**la luna**
secuencia 2	**la luna**
ratio	0.778

secuencia 1	**la luna**
secuencia 2	**la luna**
ratio	0.857

Así constatamos que diferencias en la estructura impactan en el puntaje, en definitiva, los elementos que dan forma al texto terminan siendo simplemente caracteres como cualquier otro a vistas del algoritmo. Sin lugar a dudas esta característica presenta la interrogante sobre si procesar o no el texto en términos de estructura: quitar saltos de líneas, reducir cantidad de espacios en blanco hasta quedarse con uno si existen varios contiguos, etc.

Investiguemos ahora qué ocurre cuando se tienen secuencias muy parecidas con la diferencia de que una contiene ciertos términos distintos en el medio. Resulta razonable plantearse la pregunta de si introducir "ruido" en medio puede llegar a provocar un "desfasaje" a la hora de matchear textos, el cual provoque que documentos que en realidad eran muy parecidos sean catalogados como distintos.

secuencia 1	**El día había amanecido fresco y despejado.**
secuencia 2	**El día había amanecido** ruido **fresco y despejado.**
ratio	0.933

secuencia 1	**El día había amanecido fresco y despejado.**
secuencia 2	**El día había amanecido** mucho, mucho más ruido **fresco y despejado.**
ratio	0.785

Podemos ver que el resultado es satisfactorio, el ruido es "ignorado" y las subsecuencias matcheadas no sufren ningún tipo de desfasaje anormal. ¿Pero qué ocurre si el ruido desafortunadamente contiene las palabras que debían ser matcheadas?

secuencia 1	**El día había amanecido fresco y despejado.**
secuencia 2	**El día había amanecido** mucho, mucho **fresco** más ruido **y despejado.**
ratio	0.785

secuencia 1	**El día había amanecido fresco y despejado.**
secuencia 2	**El día había amanecido** rr**fresco**rr rr**y**rr **despejado.**
ratio	0.891

Nuevamente, el resultado es bastante alentador ya que el algoritmo encuentra coincidencias que resultan naturales para el ser humano. Bajo la misma cantidad de ruido, el hecho de colocar palabras correctas en medio del mismo o acumularlo todo a cierta altura del texto no varía el puntaje final.

Finalmente, indaguemos qué ocurre a la hora de trabajar procesando textos, ¿corregir palabras realmente aumenta el puntaje? Supongamos el siguiente fragmento de texto, en el cual alrededor de la mitad de sus palabras posee al menos un error en alguno de sus caracteres.

secuencia 1	**El negro y** escarla**ta estaba enroscado en torno a sus** h**ombros, co**n **el cuello largo y sinuoso baj**o **su** bar**billa.**
secuencia 2	**El** l **negro y** zscarla**la éstaba enr**rscado **en torno a sus** b**ombros, cn el cuell**lo **largo y sinuoso baj**he **su** bra**ba**d**illa.**
ratio	0.901

Si se corrige una única palabra "*Ell*" → "*El*" obtenemos un puntaje de 0.905, donde ya se ve un avance en el puntaje. Si pasamos a corregir más palabras: "*Ell*" → "*El*", "*zscarlala*" → "*escarlata*", "*bombros*" → "*hombros*", "*brabadilla*" → "*barbilla*", "*bajhe*" → "*bajo*", "*enrrscado*" → "*enrroscado*" terminamos con un puntaje de 0.977, superando a ambos puntajes previos.

Bajo estas condiciones consideramos que la clase SequenceMatcher del módulo difflib es una herramienta que puede ser utilizada a la hora de evaluar resultados en el presente proyecto.

2.4.2. Similitud BLEU

BLEU (o *Bilingual Evaluation Understudy*) [11] es una medida desarrollada con la finalidad de evaluar la calidad de las traducciones de los sistemas de traducción automática (MT). Para calcular esta medida, se utiliza el método BLEU, que ofrece una serie de ventajas:

- Es rápido y barato de calcular.
- Fácil de entender.
- Es independiente de los lenguajes comprendidos.
- Se relaciona fuertemente con evaluaciones hechas por humanos.
- Ha sido ampliamente adoptado para la evaluación de estos sistemas.

Normalmente, hay muchas traducciones "perfectas" para una oración dada. Estas traducciones pueden variar en la elección de palabras o en el orden de las palabras, incluso utilizando las mismas palabras. Entonces, ¿cómo sabemos que las traducciones son lo suficientemente buenas como para transmitir el significado correcto? En general, nos basamos en tres conceptos: adecuación, fluidez y fidelidad de las traducciones.

- Adecuación: es una medida para saber si se expresó todo el significado desde el idioma de origen al idioma de destino.
- Fluidez: qué tan bien formadas están las oraciones gramaticalmente y la facilidad de interpretación.
- Fidelidad: es la medida en que una traducción traduce con precisión el significado del texto original.

Otro desafío en las traducciones es el uso de diferentes opciones de palabras y el cambio del orden de las palabras. A continuación, algunos ejemplos:

Diferentes opciones de palabras, pero con el mismo significado
Disfruté el concierto
Me gustó el espectáculo
Disfruté el show

Diferente orden de palabras que transmite el mismo mensaje
Llegué tarde a la oficina debido al embotellamiento
El embotellamiento fue responsable de mi retraso a la oficina.
El embotellamiento retrasó mi llegada a la oficina

Teniendo en cuenta estas complejidades, el método de BLEU propone: Cuanto más se acerque una traducción automática a una traducción humana profesional, mejor será.

Algunos términos utilizados:

- La traducción de referencia es la traducción realizada por un humano.
- La traducción candidata es la traducción automática realizada por el sistema de traducción.

De esta forma, para medir la efectividad de la traducción automática, se evalúa la cercanía de la traducción candidata a la traducción de referencia calculando la métrica conocida como *BLEU score* o puntaje BLEU. La cercanía de la traducción se mide comparando la elección y el orden de las palabras entre la traducción humana de referencia y la traducción candidata generada por la máquina.

Para entender cómo encuentra las diferencias el método se muestran algunos ejemplos.

Ejemplo 1

Candidata 1: Es una guía de acción la cual asegura que los militares siempre obedezcan las órdenes del partido.

Candidata 2: Es para asegurar que las tropas escuchen siempre la guía de actividades que dirigen el partido.

Referencia 1: Es una guía de acción que asegura que los militares siempre prestarán atención a las órdenes del partido.

Referencia 2: Es el principio rector el cual garantiza que las fuerzas armadas estén siempre bajo el mando del partido.

Referencia 3: Es la guía práctica para que el ejército siga siempre las instrucciones del partido.

A simple vista, se puede apreciar claramente que **Candidata 1** comparte muchas palabras y frases con las tres referencias, mientras que **Candidata 2** no. Las coincidencias se muestran en la tabla 2.1

Para medir esto, BLEU compara los n-gramas de la traducción candidata con los n-gramas de la traducción de referencia para contar el número de coincidencias. Estas coincidencias son independientes de las posiciones en las que ocurren. Cuanto mayor sea el número de coincidencias entre la traducción candidata y la de referencia, mejor será la traducción automática. Para esto, la métrica se basa fuertemente en la medida de precisión. Para calcular la precisión, uno simplemente cuenta la cantidad de n-gramas de las traducciones candidatas que ocurren en cualquier traducción de referencia, y luego

	Referencia 1	Referencia 2	Referencia 3
Candidata 1	es una guía de acción asegura que los militares siempre las órdenes del partido	es cual que siempre del partido	es guía la que siempre las del partido
Candidata 2	es que siempre partido	es que las siempre partido	es para siempre partido

Tabla 2.1: Tabla de coincidencias entre traducciones de referencia y candidatas

divide por el número total n-gramas en la traducción candidata, o sea si se quiere calcular la precisión unigrama, se cuenta la cantidad de palabras en la traducción candidata que ocurren en cualquier traducción de referencia, y luego se divide por el número total de palabras en la traducción candidata. Desafortunadamente, los sistemas de traducción automática pueden generar en exceso palabras "razonables", lo que resulta en traducciones improbables, pero de alta precisión, como la del ejemplo 2 a continuación.

Ejemplo 2

Candidata 1: La la la la.

Candidata 2: Es una gata en la alfombra.

Referencia 1: La gata está sobre la alfombra.

- Precisión unigrama **Candidata 1**: 4/4 (100 %)
- Precisión unigrama **Candidata 2**: 3/6 (50 %)

Ambas traducciones candidatas tienen una precisión unigrama alta, aunque sabemos que no son buenas traducciones, una no tiene sentido mientras que la otra dice qué es lo que está sobre la alfombra en vez de decir donde está la gata. Para resolver el problema, usaremos una "precisión n-grama modificada". Se calcula en varios pasos para cada n-grama.

Para entenderlo, a continuación se calcula la precisión unigrama y bigrama modificada para Candidata 1 del ejemplo 1. Primero hay que calcular $Count_{clip}$ para cualquier n-grama usando los siguientes pasos:

1. Contar el número máximo de veces que ocurre un n-grama candidato en cualquier traducción de referencia única. Se le llama *Count*.
2. Para cada oración de referencia, contar el número de veces que ocurre un n-grama candidato. Como hay tres traducciones de referencia, se calcula el *Count* de Ref 1 ($Count_{ref1}$), el *Count* de Ref2 ($Count_{ref2}$) y el *Count* de Ref 3 ($Count_{ref3}$).
3. Tomar el número máximo de ocurrencias de n-gramas en cualquier *Count* de referencia ($max(Count_{ref1}, Count_{ref2}, Count_{ref3})$). A esto se le llama $MaxRefCount$.

4. Tomar el mínimo entre *Count* y $MaxRefCount$. También conocido como $Count_{Clip}$, ya que recorta el *Count* total de cada palabra candidata por su $MaxRefCount$.

$$Count_{clip} = min(Count, MaxRefCount)$$

5. Calcular la suma de todos estos $Count_{clip}$. Se muestran los $Count_{clip}$ y el resultado de la suma de los unigramas y bigramas en la Tabla 2.2 y la Tabla 2.3 respectivamente.

6. Dividir los $Count_{clip}$ por el número total de n-gramas candidatos para obtener la puntuación de precisión modificada (p_n).

$$p_n = \frac{\sum_{C \in \{Candidatas\}} \sum_{n\text{-}grama \in C} Count_{clip}(n\text{-}grama)}{\sum_{C' \in \{Candidatas\}} \sum_{n\text{-}grama' \in C'} Count(n\text{-}grama')}$$

Count unigramas	Count	Ref1 Count	Ref2 Count	Ref3 Count	Max Ref Count	Clip Count
es	1	1	1	1	1	1
una	1	1	0	0	1	1
guía	1	1	0	1	1	1
de	1	1	0	0	1	1
acción	1	1	0	0	1	1
la	1	0	0	1	1	1
cual	1	0	1	0	1	1
asegura	1	1	0	0	1	1
que	1	2	1	1	2	1
los	1	1	0	0	1	1
militares	1	1	0	0	1	1
siempre	1	1	1	0	1	1
obedezcan	0	0	0	0	0	0
las	1	1	1	1	1	1
órdenes	1	1	0	0	1	1
del	1	1	1	1	1	1
partido	1	1	1	1	1	1
17						**16**

Tabla 2.2: Count clip unigramas Candidata 1 del ejemplo 1

Según los datos de las tablas Tabla 2.2 y 2.3 las puntuaciones de precisión de n-gramas modificada son las siguientes:

- Puntuación precisión unigrama modificada: **16/17**
- Puntuación precisión bigrama modificada: **11/16**

Count bigramas	Count	Ref1 Count	Ref2 Count	Ref3 Count	Max Ref Count	Clip Count
es una	1	1	0	0	1	1
una guía	1	1	0	0	1	1
guía de	1	1	0	0	1	1
de acción	1	1	0	0	1	1
acción la	0	0	0	0	0	0
la cual	0	0	0	0	0	0
cual asegura	0	0	0	0	0	0
asegura que	1	1	0	0	1	1
que los	1	1	0	0	1	1
los militares	1	1	0	0	1	1
militares siempre	1	1	0	0	1	1
siempre obedezcan	0	0	0	0	0	0
obedezcan las	0	0	0	0	0	0
las órdenes	1	1	0	0	1	1
órdenes del	1	1	0	0	1	1
del partido	1	1	1	1	1	1
16						**11**

Tabla 2.3: Count clip bigramas Candidata 1 del ejemplo 1

Los resultados arrojados por la precisión modificada indican en principio que la Candidata 1 es precisa, particularmente cercano a un 94 % para los unigramas y un 69 % los bigramas, aunque este no es el resultado final del puntaje BLEU.

El último detalle a tener en cuenta es el largo de las traducciones. ¿Qué pasa si las traducciones candidatas son muy cortas o muy largas? Para atacar este problema se agrega una penalización llamada Penalización por Brevedad o en inglés *Brevity Penalty* (BP).

La BP será 1.0 cuando el largo de la traducción candidata sea el mismo que el largo de cualquier traducción de referencia. Por ejemplo, si hay tres referencias de largo 12, 15 y 17 palabras y la traducción candidata es de 12 palabras, la penalización por brevedad será 1.0. Al largo de oración de referencia más cercana se le llama “largo de mejor coincidencia”.
Con la penalización por brevedad, se genera que una traducción candidata con una puntuación alta coincida con las traducciones de referencia en longitud, elección de palabras y orden de palabras.

Finalmente, BLEU se calcula como:

$$BLEU = BP \cdot exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

Siendo:

- BP: Penalización por Brevedad.
- N: Mayor número n de n-grama hasta el que se calculan coincidencias. Generalmente se usa 4, por lo tanto se calculan coincidencias entre unigrama, bigrama, trigrama y cuatrigrama.
- w_i: Peso de cada precisión modificada. Generalmente se usa 1/4.
- p_n: Precisión modificada.

La puntuación BLEU varía de 0 a 1. Cuando la traducción automática es idéntica a una de las traducciones de referencia, se obtiene una puntuación de 1.

El puntaje BLEU de las traducciones candidatas del ejemplo 1 respecto a las tres referencias (utilizando N=4 y w_i = 1/4) es:

- Candidata 1: 0.558662320386.
- Candidata 2: 0.0 (No tiene trigramas ni cuatrigramas en común con ninguna referencia).

2.5. Trabajos previos sobre corrección de textos generados por OCR

Históricamente siempre ha habido una investigación considerable en el campo de la corrección automática de textos en general, y particularmente la corrección de textos generados por OCRs no es ajeno a ello.

El problema de mejorar el trabajo de los OCRs puede atacarse por tres sectores distintos. Se puede buscar mejorar la entrada brindada al OCR, el sistema de OCR en sí, o la salida de texto generada. Este proyecto se enfocó mayoritariamente en el último punto por lo que daremos una breve reseña sobre trabajos que toman a los OCR como caja negra y han indagado principalmente en la corrección de los textos una vez han sido producidos.

Se han explorado diversos enfoques entre los que se incluyen aprendizaje automático (tanto supervisado como no supervisado). Entre ellos, algunos pueden funcionar como herramientas auxiliares que brindan ayuda adicional a correctores humanos como es el caso del trabajo de Abdulkader et al. [12]. En este proyecto se construyó una red neuronal que aprende a reconocer errores devolviendo una probabilidad de qué tan factible es que la palabra estudiada sea error o no. Luego dependiendo del valor de esa probabilidad se decide si un humano debe revisar la palabra. Por otro lado, otros enfoques funcionan de forma independiente con respecto a correctores humanos, corrigiendo de manera completamente automática los textos generados por OCRs. Como ejemplo de este último caso encontramos el trabajo de Xiang Tong et al. [13] en el cual se describe un sistema para corrección de errores basado principalmente en modelos de lenguajes. El *pipeline* de trabajo aquí empleado suele ser común entre sistemas de este estilo, en el que primero se ejecuta algún tipo de tokenizador, posteriormente cada token es evaluado mediante un módulo capaz de detectar errores, luego para cada error

detectado se proponen candidatos para corregirlos y finalmente se ejecuta un último algoritmo para decidir el mejor candidato. Una solución de este tipo fue explorada en el presente proyecto.

Otra alternativa que se ha investigado consiste en la unión (también conocido en inglés como *merge*) de las salidas generadas por más de un OCR al leer una misma imagen. Este tipo de solución es explorada por ejemplo en el trabajo de Volk et al. [14]. En este se buscó tomar ventaja de la premisa de que distintos sistemas de OCR generan distintos errores, y que idealmente se podría lograr seleccionar las "partes correctas" de cada texto generado.

Por otro lado, Kolak et al. [15] presentan un sistema conformado por una máquina de estado finito que trabaja a nivel de caracteres. Están interesados en mejorar el nivel de los OCRs en nuevos lenguajes aparte del inglés mostrando que un OCR para el idioma inglés puede producir buenos resultados para otro idioma cuando se combina con una herramienta de alineación a nivel de palabras.

Siguiendo con esta línea, se han implementado diversas soluciones bajo la utilización de sistemas de traducción automática (SMT) como correctores, donde la idea principal consiste en entrenar al modelo para que aprenda a traducir desde el lenguaje con errores producido por el OCR hacia el mismo lenguaje ya corregido. Afli et al. [16] construyen un sistema de traducción y lo comparan contra un sistema de corrección constituido por un modelo de lenguaje similar al visto en [13]. Los resultados del experimento arrojaron que el sistema de traducción superó a la corrección por medio del modelo de lenguaje, siendo este trabajo quien diera el puntapié inicial para que en este proyecto explorásemos la corrección de textos por medio de sistemas de traducción automática.

Como última mención sobre trabajos relacionados, encontramos que es común la implementación de sistemas que combinen más de una estrategia o módulo de corrección. Como ejemplo de esto podemos poner a [16] nuevamente, ya que además de comparar el SMT con un modelo de lenguaje, diseñaron dos sistemas adicionales en los que intervienen ambos módulos en conjunto. La diferencia entre estos dos sistemas es el orden en el que se ejecutan los módulos.

Capítulo 3

Ground truth

El término *ground truth* se traduce como la verdad sobre el terreno o dicho de otra forma la verdad obtenida en la práctica. Es el resultado al que se quiere llegar.

El *ground truth* será utilizado para poder comparar con algo que se considera correcto, sirve para poder medir los resultados obtenidos, ajustar los parámetros de los procesamientos y por último evaluar las soluciones planteadas.

En el caso de nuestro proyecto el *ground truth* se construye con los datos obtenidos del proyecto LUISA. Se toma esta decisión porque los datos son una transcripción manual de los documentos, y en este caso son más fiables que las transcripciones generadas por alguna herramienta de procesamiento de imágenes debido a diversos factores. Entre ellos la experiencia previa del traductor, el entendimiento del contexto temporal y social entre muchos otros.

3.1. Origen de los datos

En un capítulo anterior se ha comentado del proyecto comunitario denominado LUISA, este proyecto que precisa la participación de la comunidad para realizarse busca generar transcripciones manuales de cada palabra o frase de los documentos obtenidos.

Por lo tanto, el grupo encargado del proyecto tiene en su posesión documentos escaneados de microfilm en formato ".tiff", imágenes de palabras recortadas del documento escaneado llamadas bloques y las transcripciones de estos recortes realizadas por la comunidad. A estos datos pudimos acceder y son los que utilizamos para el proyecto.

Contamos con la primera tanda de documentos llamada "tanda0" y la segunda llamada "tanda1". La tanda cero por su parte consta de 163 documentos, que suman 58.375 bloques y 1.223.055 transcripciones. En promedio cada documento tiene 158 bloques. Los bloques no son necesariamente una palabra, a veces constan de un conjunto reducido de palabras o partes de palabras, otras veces contienen manchas u otras imperfecciones, también se tiene que tener en cuenta que para un bloque existen muchas transcripciones que no necesariamente son idénticas, algunas incluso pueden ser vacías.

Por su parte, la segunda tanda contiene 1000 documentos, sumando 440.165 bloques y 2.171.045 transcripciones, lo que da un promedio de 440 bloques por documento.

Los datos fueron compartidos como documentos enteros escaneados en formato ".tiff" y de un *dump* de base de datos. Este *dump* de base de datos contiene tablas con información de las hojas, bloques

y textos (transcripciones). Los bloques son los recuadros que se presentan a la comunidad para que transcriban, las transcripciones de estos bloques son los textos. Se destaca que como dato del bloque se encuentra las coordenadas en la hoja de dos puntos del bloque, el punto superior izquierdo y el punto inferior derecho, con esos dos puntos se puede obtener el rectángulo de imagen a la que pertenece el bloque.

Un ejemplo de una parte de un documento se puede apreciar en la Imagen 3.1. Esta porción de documento contiene varios bloques, uno de ellos está formado por la palabra "Río".

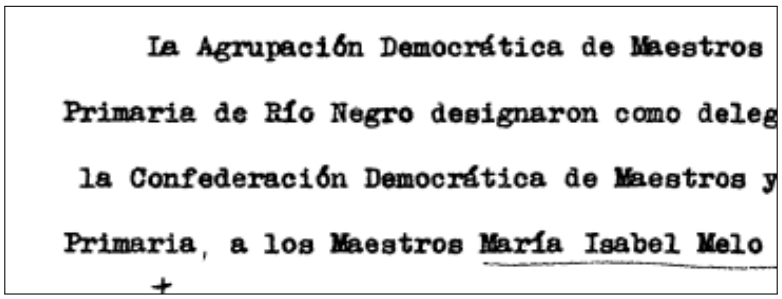


Imagen 3.1: Extracto documento escaneado

Utilizando las coordenadas del bloque de la palabra mencionada, se puede obtener su recorte (Imagen 3.2) en el documento escaneado.

Imagen 3.2: Bloque recortado

Los bloques pueden ser transcritos de distintas maneras según las interpretaciones de cada persona, para el caso de la palabra "Río" se tiene que fue transcrita ocho veces como "Río" y una vez como "Rio" (sin tilde) por distintos usuarios.

Este será el input para la reconstrucción de los documentos.

3.2. Construcción

Se generaron dos instancias de *ground truth*, cada una con una aplicación distinta dentro del proyecto:

- Parejas imagen-bloque: es una colección de pares de bloques recortados y sus respectivas transcripciones. Este conjunto especifica una relación entre imágenes recortadas de los documentos digitales y sus transcripciones asociadas obtenidas de la plataforma LUISA. De las distintas transcripciones posibles para un bloque, se decide por la mejor opción. Dicho *ground truth* fue utilizado para entrenar Tesseract (uno de los OCRs estudiados).
- Reconstrucciones completas de documentos: esta segunda instancia de *ground truth* consta de la reconstrucción completa de documentos. Esto significa el rearmado línea por línea de cada documento utilizando las mejores opciones de transcripción para cada bloque del documento original, al igual que en parejas imagen-bloque. El objetivo de dicha reconstrucción es lograr obtener un texto que simule haber sido transcrito por una persona.

Parejas imagen-bloque

Para generar el primer *ground truth* se procedió a reconstruir los bloques a partir de los datos presentes en la base de datos. Cada bloque está definido por dos puntos que representan su esquina superior izquierda y esquina inferior derecha dentro del documento. Estos puntos a su vez están determinados por una pareja de coordenadas (x, y), siendo x la cantidad de píxeles a moverse en el ancho del documento e y el número de píxeles en relación al alto del mismo. En la Imagen 3.3 se puede apreciar un ejemplo de un bloque el cual encierra a la palabra "apelaciones". En la misma se marcaron de modo ilustrativo los dos puntos que definen al bloque. Una vez establecidas las coordenadas de un bloque, se prosigue con el recorte de este del documento utilizando la librería de Python llamada "PIL". Esta librería recibe las coordenadas de un rectángulo y genera una imagen con el recorte del documento. Aplicando este procedimiento para todos los bloques se obtienen imágenes con el contenido de cada bloque aisladas del resto del documento.

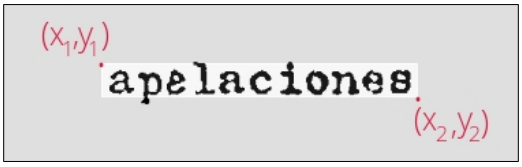


Imagen 3.3: Coordenadas correspondientes a un bloque

Para obtener la palabra asociada a cada bloque, se toma en cuenta el ID del bloque y se buscan todas las transcripciones asociadas a este ID. Estas transcripciones se ordenan según la cantidad de ocurrencias para cada bloque: si un bloque recibió 3 transcripciones desde los usuarios de LUISA, donde dos de ellas coinciden y una tercera no, la transcripción que aparece 2 veces irá primera en el ordenamiento. Si existe una única transcripción con mayor cantidad de ocurrencias, entonces se asociará al bloque. Si existen dos o más transcripciones con mayor cantidad de ocurrencias se hará uso de un vocabulario generado para el proyecto para desempatar, este vocabulario será presentado en un capítulo siguiente. Para desempatar las transcripciones con más ocurrencias, se recorrerán las mismas y se seleccionará la primera que pertenezca al vocabulario. En caso de que ninguna pertenezca al vocabulario, simplemente se seleccionará la primera de ellas.

En la tabla 3.1 podemos ver algunos ejemplos con las transcripciones que se asociaron a 3 bloques distintos. En la primera y tercera tabla no fue necesario usar el diccionario dado que no ocurrió un empate en mayor cantidad de ocurrencias.

Palabra	Frec.	Vocab.
patrulló	4	Si
patrulla	2	Si
patrullo	2	Si

Palabra	Frec.	Vocab.
detenido-	2	No
detenida	2	Si
detenido	2	Si

Palabra	Frec.	Vocab.
Cufré	4	No
Cufre	1	No

Tabla 3.1: Ejemplos de transcripciones, en rojo las elegidas

Reconstrucciones completas de documentos

Para generar la segunda instancia de *ground truth* se comienza recorriendo las hojas. Para cada hoja se obtienen los bloques ordenados de forma decreciente. Esto es, si tomamos como referencia a la hoja, obtenemos una lista de sus bloques ordenados de arriba hacia abajo y de izquierda a derecha utilizando las coordenadas x e y.

Luego se asocia la transcripción de cada bloque de forma análoga a como se hizo en el conjunto de parejas imagen-bloque. Finalmente, los documentos se reconstruyen uniendo las palabras asociadas a cada bloque de manera ordenada, insertando espacios entre cada uno. De esta forma se logra ir armando el documento bloque por bloque.

Un detalle a tener en cuenta son los saltos de línea a la hora de reconstruir el documento palabra por palabra. Para interpretar que entre dos bloques se debe insertar un salto de línea se compara las posiciones entre un bloque y el siguiente. Se inserta un salto de línea cuando se cumple la siguiente condición: la coordenada y del punto inferior derecho de un bloque (esta coordenada indica la ubicación de la arista inferior del bloque en el documento) es menor a la coordenada y del punto superior izquierdo de bloque siguiente. O, dicho en otras palabras, el primer bloque está por encima del bloque siguiente.

	Bloq. id	Palabra frec.	y_0	y_1
1.	194643	indagado	2556	2596
2.	194644	por	2556	2596
3.	194645	distribuir	2556	2596
4.	194646	"billetes"	2616	2656
5.	194647	ofensivos	2616	2656
6.	194648	al	2616	2656
7.	194649	actual	2616	2656
8.	194650	gobierno,	2616	2656
9.	194651	a	2616	2656
10.	194652	disposición	2616	2656
11.	194653	de	2740	2780
12.	194654	la	2740	2780
13.	194655	Justicia	2740	2780

Tabla 3.2: Ejemplo de bloques con parte de sus coordenadas y transcripciones, los colores en las coordenadas representan las distintas líneas reconocidas.

En la Tabla 3.2 se recorren las palabras frecuentes y se concatenan con un espacio entre medio. Cuando se detecta un salto de línea entre dos bloques consecutivos se lo agrega a la reconstrucción. De esta forma, con los bloques anteriores se obtienen las líneas de la tabla 3.3.

	Transcripción
1.	indagado por distribuir
2.	”billetes” ofensivos al actual gobierno, a disposición
3.	de la Justicia

Tabla 3.3: Ejemplo de reconstrucción de documento

Una vez reconstruidos todos los documentos se tiene un *ground truth* listo para utilizar en las partes siguientes. Se puede ver un pseudocódigo del algoritmo de reconstrucción en la sección "Pseudocódigo de las reconstrucciones" del anexo.

3.3. Dificultades al reconstruir documentos

La reconstrucción de los bloques puede presentar dificultades, algunas salvables y otras no. A continuación se presentan algunas de las dificultades encontradas.

Palabras separadas

Analizando los resultados de la reconstrucción y los datos de LUISA se pudo identificar que en muchos casos una misma palabra estaba separada en varios bloques. Bajo el algoritmo de reconstrucción planteado anteriormente, el resultado de procesar esta palabra son varios fragmentos de palabra separados por espacios. Por ejemplo, si se tiene la palabra "indagados" dividida en cuatro bloques "in", "dag", "ad" y "os", se reconstruiría como "in dag ad os". Esto representa un problema para las etapas de corrección y evaluación dado que si se intentara comparar o evaluar usando las partes por separado se cometería un error. Como los documentos están escritos en máquina de escribir, los espacios entre letra y letra pueden no ser uniformes, esto genera que, al escanear los documentos, una herramienta de procesamiento de imágenes puede interpretar la existencia de espacios entre letras que no lo tenían. El efecto anterior es el que genera múltiples bloques para una palabra.

Para solucionar esto se investigó la relación entre los bloques contiguos con muy poca separación horizontal entre sí. Se estudiaron los conjuntos de bloques que se encuentran a una distancia horizontal entre 1 y 30 pixeles, se los agrupó y clasificó en cuatro categorías: **wrongs**, **undefined_wrongs**, **undefined_corrects** y **corrects**. Las categorías se definen según la certeza que se tenga de que la unión de los bloques contiguos sea correcta.

- **wrongs**: conjuntos de bloques contiguos en los cuales alguna de sus partes pertenece al vocabulario, pero la unión de estas no.
- **undefined_wrongs**: conjunto de bloques contiguos en los cuales tanto sus partes separadas y su unión no pertenecen al vocabulario.
- **undefined_corrects**: conjunto de bloques contiguos en los cuales alguna de sus partes separadas y la unión de estas pertenecen al vocabulario.
- **corrects**: conjunto de bloques contiguos en la cual las partes por separado no pertenecen al vocabulario, pero la unión de las misma si pertenece.

A continuación se presentan ejemplos de cada conjuntos.

Conjunto	Partes (Pertenece al Voc.)	Partes unidas
Wrongs	JUNTA + DE +COMANDANTES	JUNTADECOMANDANTES
Undefined wrongs	REZ + ZANO	REZZANO
Undefined corrects	si + ndicato	sindicato
Corrects	averi + guación	averiguación

Tabla 3.4: En verde si la palabra pertenece al vocabulario y en rojo en el caso contrario

La Tabla 3.5 presenta la cantidad de conjuntos de bloques cercanos obtenidos para cada distancia en píxeles tomada.

Dist.	Corr.	U. Corr.	U. Wrg.	Wrg.
1px	1.185	3.592	5.115	18.845
2px	1.185	3.592	5.115	18.845
3px	1.185	3.592	5.115	18.845
4px	1196	4.150	6.068	32.386
5px	1196	4.150	6.068	32.386
6px	1196	4.150	6.068	32.386
7px	1196	4.150	6.068	32.386
8px	1.041	3.947	6.606	41.214
9px	1.041	3.947	6.606	41.214
10px	1.041	3.947	6.606	41.214
11px	1.041	3.947	6.606	41.214
12px	622	2.666	6.228	43.490
13px	622	2.666	6.228	43.490
14px	622	2.666	6.228	43.490
15px	622	2.666	6.228	43.490

Dist.	Corr.	U. Corr.	U. Wrg.	Wrg.
16px	363	1.455	5.750	38.121
17px	363	1.455	5.750	38.121
18px	363	1.455	5.750	38.121
19px	363	1.455	5.750	38.121
20px	259	1.086	5.425	35.341
21px	259	1.086	5.425	35.341
22px	259	1.086	5.425	35.341
23px	259	1.086	5.425	35.341
24px	211	961	6.105	34.581
25px	211	961	6.105	34.581
26px	211	961	6.105	34.581
27px	211	961	6.105	34.581
28px	213	954	8.613	34.709
29px	213	954	8.613	34.709
30px	213	954	8.613	34.709

Tabla 3.5: Palabras agregadas en cada paso

Observando las palabras formadas por la unión de los bloques contiguos cercanos se determinó que para todos los casos de palabras clasificadas como **corrects** y clasificadas como **undefined_correct** donde no todas las partes pertenecen al vocabulario, tenía sentido juntarlas en la transcripción final, dado que casi siempre era la decisión correcta. La decisión de no considerar los bloques contiguos cercanos donde tanto las partes como la palabra unida pertenecen al vocabulario tiene intención de evitar juntar partes como “a” y “la” para formar “ala”.

En un principio se decidió únicamente utilizar la distancia de 3px para juntar bloques debido a que era la distancia que mayor cantidad de palabras juntaba. Pero luego, observando las palabras formadas por distancias mayores, se apreció muchas palabras que con 3px no llegaban a formarse. Por lo tanto, una mejor solución era utilizar también las distancias mayores. Las distancias utilizadas fueron 3px, 4px,8px, 12px, 16px, 20px y 24px, con más de 24px muchas de las palabras formadas no eran correctas.

Cabe destacar que las palabras formadas utilizando mayores distancias no incluyen a todas las que se generan utilizando menores distancias, con más distancia se pueden unir más bloques y no formar una palabra dentro del vocabulario que con una menor distancia. A continuación se explica con un ejemplo.

Si se tiene la siguiente frase:

El ho mbre fue dete nido por la pol icía.

El| $|^{20px}$ho| $|^{8px}$mbre| $|^{30px}$fue| $|^{24px}$dete| $|^{16px}$nido| $|^{24px}$por| $|^{40px}$la| $|^{40px}$pol| $|^{24px}$icía.

En rojo se marca el tamaño en píxeles de los espacios. En la tabla Tabla 3.6 se describen el conjunto de bloques reconocidos para cada distancia. Se llamará conjunto de bloques válidos al conjunto de bloques contiguos cuya unión es considerada correcta.

Distancia	Conjunto de bloques reconocidos	Conjunto de bloques válidos
24px	(El,ho,mbre),(fue,dete,nido,por),(pol,icía)	(pol,icía)
20px	(El,ho,mbre),(dete,nido)	(dete,nido)
16px	(ho,mbre),(dete,nido)	(ho,mbre),(dete,nido)
12px	(ho,mbre)	(ho,mbre)
8px	(ho,mbre)	(ho,mbre)
4px		
3px		

Tabla 3.6: Palabras reconocidas por cada distancia tomada

El método para utilizar varias distancias consiste en comenzar formando palabras desde las distancias mayores hacia las menores. Primero se obtienen todas las palabras a unir con la distancia de 24px y se marcan los bloques de las partes unidas para no volverlos a utilizar cuando se procesen distancias menores. Luego se obtienen las palabras a unir con distancia 20px y cuyas partes no fueron marcadas anteriormente. Este proceso se aplica para todas las distancias utilizadas, con lo que se obtienen todas las palabras a juntar utilizando las distancias elegidas.

Utilizando el ejemplo propuesto anteriormente y el método de considerar primero las distancias mayores se tiene la siguiente evolución del conjunto de palabras a unir representado en la Tabla 3.7.

Paso	Dist.	Conj. de bloques válidos	Partes a unir	Cant. palabras a unir
1.	24px	(pol,icía)	(pol,icía)	1
2.	20px	(dete,nido)	(dete,nido),(pol,icía)	2
3.	16px	(ho,mbre),(dete,nido)	(ho,mbre),(dete,nido),(pol,icía)	3
4.	12px	(ho,mbre)	(ho,mbre),(dete,nido),(pol,icía)	3
5.	8px	(ho,mbre)	(ho,mbre),(dete,nido),(pol,icía)	3
6.	4px		(ho,mbre),(dete,nido),(pol,icía)	3
7.	3px		(ho,mbre),(dete,nido),(pol,icía)	3

Tabla 3.7: Palabras reconocidas por cada distancia tomada

En la Tabla 3.8, se puede apreciar la evolución de la cantidad de palabras a unir en cada paso realizado partiendo desde la distancia 24px hasta la distancia 3px sobre la totalidad de los documentos.

Paso	Distancia	Conj. de bloques válidos	Palabras agregadas	Cant. palabras a unir
1.	24px	491	491	491
2.	20px	608	152	643
3.	16px	946	365	1.008
4.	12px	1.894	991	1.999
5.	8px	3.047	1.253	3.252
6.	4px	3.421	620	3.872
7.	3px	3.164	318	4.190

Tabla 3.8: Palabras agregadas en cada paso para la totalidad de documentos

Luego de los siete pasos anteriores se identificaron sobre todo el corpus 4.190 palabras que estaban separadas en varios bloques contiguos y en el momento de la reconstrucción de documentos se procedió a juntarlas.

Calidad de Escaneos

Una de las dificultades está relacionada con la calidad de escaneo de los microfilms y los bloques identificados por la herramienta de procesamiento de imagen utilizada en el proyecto LUISA. En muchos casos al escanear el microfilm queda marcado el borde de este como si fuera un marco, esto genera una mancha que no tiene que ver con el documento original. Dicha mancha es reconocida en muchas oportunidades como un bloque con texto y puesta a disposición para que la comunidad la transcriba. Este suceso puede generar ruido en el resultado final de la transcripción cuando se reconstruye el documento, dado que entre bloques de palabras se puede encontrar bloques originados por manchas. Un ejemplo de documento manchado por el escaneo se aprecia en la Imagen 3.4.

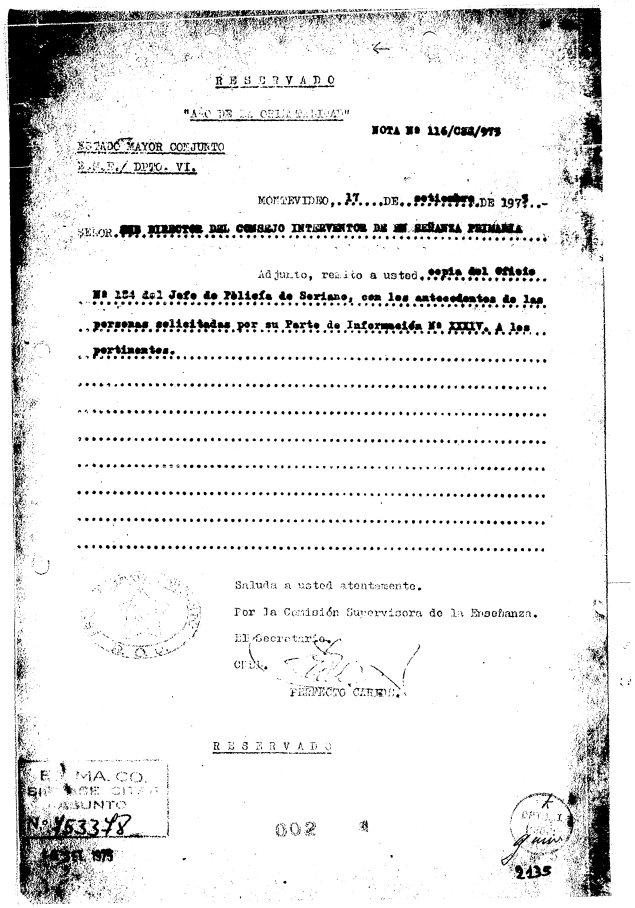


Imagen 3.4: Documento con manchas en el margen

Mala transcripción de bloques

Otra dificultad encontrada ocurre cuando al caso anterior de identificación de manchas ajenas al documento original, se le suma un mal entendimiento de la comunidad al transcribir lo que observa en el proyecto LUISA. Un ejemplo se observa en el bloque recortado de la Imagen 3.5 obtenida del documento de la Imagen 3.4. Para el bloque mencionado existen nueve transcripciones ocho de ellas son vacías mientras que la restante tiene la siguiente frase “Orillo manchado de documento”. Debido a que las transcripciones vacías se filtran, la única transcripción válida es la mencionada anteriormente, se puede apreciar que no es una transcripción sino una descripción de lo observado. Por lo tanto, es ruido que se agrega al *ground truth*. Debido a la similitud entre una descripción y una palabra trascripta válida (ejemplo “mancha”) y a la poca frecuencia con la que se observó que ocurrían, se tomó

la decisión de no intentar corregirlas.

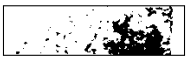


Imagen 3.5: Bloque mal identificado

Formatos de documentos

Se identificó una dificultad que tiene su origen en el formato de los documentos y en su alineación. Los documentos no son únicamente párrafos de texto, sino que estos pueden presentarse como tablas, cartas, listas o formularios entre otros. Podemos ver algunos ejemplos de estos formatos en la imagen 3.7.

En el caso de un formato tipo tabla, al reconstruir no se identifican los espacios y el alineamiento de columnas se pierde, quedando un texto difícil de comprender. Una posible solución podría ser utilizar las coordenadas de los bloques para asignar espacios, o construir una matriz con las coordenadas. Para utilizar esta solución primero se debe resolver el problema de superposición horizontal de bloques, esto se podría hacer en un trabajo futuro.

En los documentos tipo formulario o planilla se genera mucho ruido producto de bloques mal identificados con las líneas que dividen los campos, esas líneas pueden ser continuas o punteadas. En la etapa de procesamiento de documentos la aparición de caracteres producto del formato del documento inferirá en el resultado del procesamiento.

Sellos en documentos

Una de las dificultades que pueden ser salvables es la aparición de sellos en los documentos. Los sellos pueden ser reconocidos como palabras por la herramienta de procesamiento de imagen utilizada para LUISA, esto genera que sean puestos a disposición de la comunidad para transcribir. El sello más común que aparece en casi todos los documentos es el de la Imagen 3.6.

Imagen 3.6: Sello más común

No es buena idea mantener sellos dado que estos aparecen en la mitad de los textos y al momento de transcribir pueden aparecer entre frases. Para filtrar el sello mencionado durante la transcripción se procede a evitar reconstruir las palabras DEP., I, y PROCESADO si ocurren cerca entre sí.

Para utilizar el *ground truth* se tiene que tener en cuenta que estas dificultades pueden influir en el mismo, y por lo tanto se debe intentar minimizar su impacto al considerarlo como la verdad.

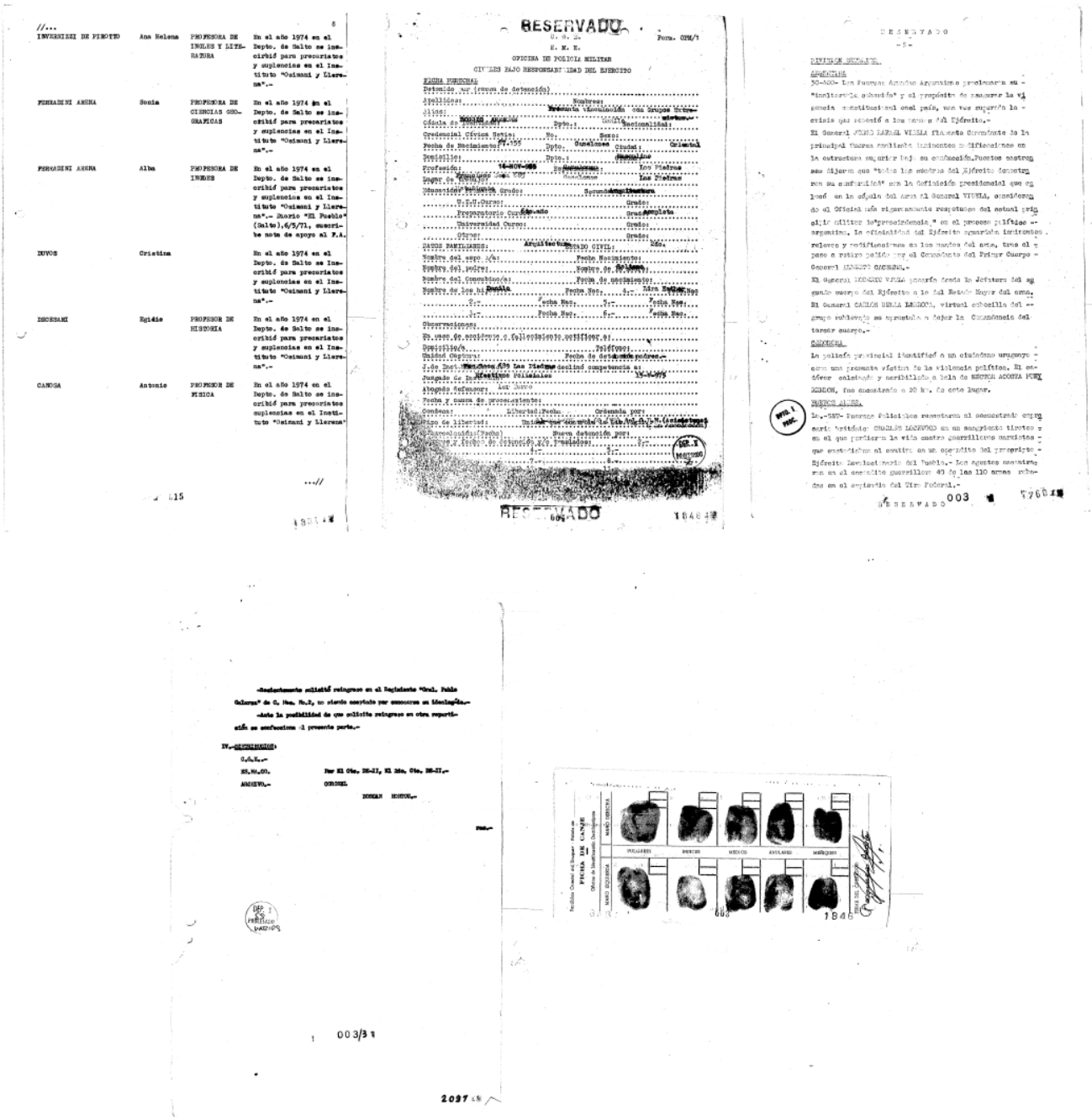


Imagen 3.7: Formatos de documentos

3.4. Evaluación de las reconstrucciones

Con el objetivo de medir el desempeño de las reconstrucciones, se evaluó el nivel de similitud entre estas y las transcripciones manuales de los documentos asociados. Es así que el equipo transcribió 15 documentos a partir de las imágenes escaneadas para ser utilizados en la comparación.

La similitud se midió utilizando la función *ratio* explicada más en detalle en el capítulo 2.4. De todas formas para entender mejor los resultados que veremos a continuación, es necesario dar una breve introducción a la herramienta de evaluación: *ratio* recibe dos textos y devuelve un puntaje representado por un número racional entre 0 y 1 que denota el nivel de similitud entre los textos dados. Cuanto más cercano a 1 es este resultado, más similares son los textos y, por lo contrario, valores bajos indican mayores diferencias. La documentación de la herramienta aclara que puntajes de 0.6 o más, reflejan textos con un alto nivel de similitud.

En la tabla 3.9 podemos ver el puntaje de *ratio* obtenido para cada uno de los 15 ejemplos.

Ejemplo	Puntaje
1	0.95736
2	0.84754
3	0.84118
4	0.92543
5	0.94404
6	0.95014
7	0.78953
8	0.84998
9	0.89711
10	0.80938
11	0.77637
12	0.96779
13	0.84243
14	0.72822
15	0.94454

Tabla 3.9: Similitud entre reconstrucciones y transcripciones manuales

Como se puede apreciar, los puntajes en general son de buenos a muy buenos. El promedio de todos ellos es de 0.87140, resultado alentador considerando que 0.6 es una medida aceptable. Los puntajes más altos se dieron en los documentos más claros, con menor cantidad de símbolos y manchas. A modo de conclusión, entendemos que tomar como *ground truth* las reconstrucciones a partir de los datos de LUISA es una decisión razonable respaldada no solo por la intuición y la percepción de que los textos generados eran buenos, sino también por los experimentos aquí realizados.

Capítulo 4

Conversión de imágenes a texto

En esta sección se presentan las herramientas estudiadas para llevar a cabo la transcripción de las imágenes, las posibilidades de entrenamiento de cada una, y la comparación de las mismas. Al final del capítulo se encuentran los resultados de la comparación mostrando que el uso de la herramienta Abbyy Fine Reader 14 es la elección óptima para el problema.

4.1. Herramientas para OCR

Un OCR u *Optical Character Recognition* es una herramienta de análisis, cuyo objetivo es el de escanear un documento con texto como un pdf o una hoja escaneada y generar un archivo que puede ser leído y editado por un procesador de texto.

Los OCRs son capaces de reconocer caracteres tanto escritos en computadora o máquina de escribir como letra cursiva o manuscrita. Según la configuración o el tipo de OCR pueden ser capaces de reconocer distintos idiomas. Además de la diversidad de idiomas, algunos permiten ser entrenados para poder interpretar textos más eficazmente.

Se decidió estudiar y comparar tres herramientas: Tesseract 4, ABBYY Fine Reader 14 y Readiris 17. Esta decisión fue determinada principalmente por dos razones: la posibilidad de entrenamiento que brindan y el conocimiento de que se encuentran entre las herramientas más utilizadas para proyectos de la misma índole. Una limitante importante a la hora de definir qué aplicaciones evaluar fue la de no poder utilizar una solución en la nube, debido a la confidencialidad de los datos.

4.1.1. Tesseract 4

Tesseract [17] es un motor de OCR de código abierto bajo licencia Apache. Fue desarrollado por Hewlett-Packard a mediados de la década de los 80's y liberado como software libre en el 2005, desde 2006 es mantenido por Google. En ese mismo año llegó a ser considerado uno de los OCRs más precisos.

Tesseract está desarrollado para múltiples sistemas operativos como Windows, Linux y MacOS, aunque testeado rigurosamente únicamente para los dos primeros. Originalmente no presenta ninguna interfaz gráfica, pero existen proyectos que implementan una (VietOCR [18] y OCRFeeder [19]).

Desde su versión inicial solo para idioma inglés, se le han ido agregando nuevas características como soporte para distintos formatos de entrada, nuevos idiomas, abecedarios distintos, soporte de texto

con sentido derecha a izquierda entre otros. En la última versión hasta el momento, la versión cuatro, se agregó un motor de OCR basado en una red LSTM y modelos de lenguaje para múltiples idiomas, llegando a un total de 116. Tesseract puede ser entrenado para reconocer otros idiomas, esta capacidad será utilizada para intentar mejorar los resultados obtenidos escaneando documentos de baja calidad. A continuación se explicará este proceso.

Entrenamiento

Entrenar Tesseract es posible, pero es algo complejo debido a la poca y errónea documentación. Luego de distintas pruebas sin resultados exitosos, se encontró el proyecto Tesstrain [20], el cual tiene la finalidad de facilitar el entrenamiento de Tesseract 4 mediante un *makefile* que realiza la instalación de todas las dependencias necesarias, así como también ejecutar cada uno de los pasos para llevar a cabo el entrenamiento. Para poder utilizarlo se tuvo que actualizar la versión Linux Ubuntu de 14.04 a 16.04.

Luego de descargar el proyecto Tesstrain e instalar Tesseract, se debe instalar Leptónica [21], una librería de código abierto con una gran cantidad de herramientas para el procesamiento y análisis de imágenes que es utilizada por Tesseract.

A continuación, se deben preprocesar los datos de *ground truth* para filtrar los posibles elementos que generen ruido. Recordamos que el *ground truth* a utilizar consiste en imágenes recortadas o bloques de documentos los cuales tienen su transcripción realizada por la comunidad mediante la plataforma LUISA.

En la documentación se menciona que es favorable que los bloques o imágenes utilizadas tengan al menos 30px de margen blanco cuando se va a entrenar, para agregar este margen a todos los bloques se utilizó una herramienta de edición de imágenes de uso local llamada ImageMagick [22].

En cuanto a las transcripciones se descartan los pares de bloque/trascripción cuya transcripción sea vacía o contenga el carácter "@", este carácter indica que el bloque es ilegible. La transcripción vacía no aporta información por lo que no ayuda en el entrenamiento.

Por otro lado, también son descartados los pares de bloque/trascripción con menos de un 90 % de coincidencia, dado que algunos bloques pueden ser confusos incluso para el ojo humano, se consideró que las transcripciones que no coinciden lo suficiente tienen gran probabilidad de ser erróneas. Otra restricción aplicada se da en la cantidad de usuarios distintos que transcribieron un bloque, se filtran las transcripciones realizadas por menos de tres usuarios con el mismo objetivo que la restricción anterior. Por último, se filtran las transcripciones con menos de tres caracteres o que no contengan caracteres alfanuméricos. Todas estas restricciones o filtros sobre las trascripciones se fueron ideando al estudiar los resultados obtenidos al intentar entrenar.

En un primer acercamiento al entrenamiento, luego de un trabajoso entendimiento y configuración de Tesstrain, se probó con un ejemplo simple. Se procedió a intentar mejorar la salida obtenida por Tesseract para un único bloque, el bloque con la palabra "detenido" originalmente era reconocido por Tesseract como "deteter". Entrenando únicamente con la imagen del bloque de la palabra en cuestión se comprobó exitosamente una mejora del resultado dado que se obtuvo la palabra correcta.

El siguiente paso fue utilizar directamente todos los bloques a disposición para entrenar. Esto ocurrió al inicio del proyecto y para ese momento únicamente se contaba con los bloques de la tanda 0. Teniendo Tesseract entrenado con todos los bloques disponibles se procedió a utilizarlo para escanear un documento entero. El resultado fue muy malo, significativamente peor que Tesseract sin entrenar.

Con este entrenamiento, Tesseract reconoció casi todas las palabras como conjuntos de puntos y guiones.

Se probó reduciendo el conjunto de bloques utilizados para el entrenamiento, pero no mejoró el resultado. Esto puede ser debido a la calidad de los datos, especialmente datos borrosos o manchas que logran empeorar el reconocimiento del OCR, como también puede deberse a limitaciones de la herramienta.

Como una posible mejora se podría utilizar una mayor cantidad de datos y hacer una limpieza de ellos para evitar que manchas o malas transcripciones empeoren los resultados.

A pesar de todo el tiempo invertido en entrenar Tesseract, no se obtuvo una mejora de rendimiento con el conjunto disponible. Por lo tanto, se descarta la idea de utilizar Tesseract entrenado.

4.1.2. ABBYY Fine Reader 14

ABBYY Fine Reader [23] es uno de los OCRs más conocidos y usados para el reconocimiento de documentos antiguos, que además presenta la posibilidad de reconocer varios idiomas. Otra de las cualidades que posee y que fue determinante para descartar otros OCRs es la posibilidad de utilizarlo *on-premise* y no tener que subir a la nube ningún documento. Para su uso ilimitado se requiere comprar una de las licencias que ofrecen, pero existe la posibilidad de acceder a una versión de prueba de treinta días o cien documentos escaneados.

En la documentación de la herramienta se indica el funcionamiento para reconocer un documento. Como primer paso analiza la estructura del documento y lo divide en elementos como texto, tablas, imágenes u otros elementos reconocibles.

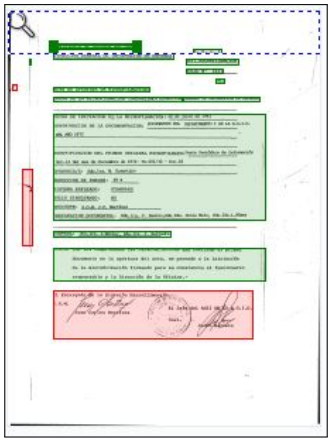


Imagen 4.1: En verde el texto reconocido y en rojo los sellos e imágenes

A las líneas de texto las divide en palabras y luego en caracteres. Cuando se tiene los caracteres señalados, el programa los compara con un conjunto de imágenes del patrón y obtiene probabilidades de coincidencia. En base a variaciones y probabilidades el programa va seleccionando caracteres, generando palabras y formando las distintas líneas. Luego de procesar un gran número de probabilidades se toma la decisión y se presenta el texto reconocido.

Entrenamiento

Es importante mencionar que el entrenamiento de Abbyy consiste en generar una lista de imágenes de caracteres junto a su respectiva transcripción la cual posee una cota superior de 1000 elementos, por lo que evidentemente existe un límite en la capacidad de este proceso, al cual se llega relativamente rápido (entrenando con 3 imágenes se definieron alrededor de 700 elementos). Este entrenamiento se realiza a través de la interfaz gráfica de la herramienta en la cual se recibe recortes de los caracteres con baja probabilidad de reconocimiento para Abbyy y el usuario ingresa su transcripción. El entrenamiento en Abbyy, por la limitante de elementos y la forma de entrenar, dio la idea de tener la finalidad de reconocer los caracteres usados específicamente en la imagen con la que se entrenó o la de reconocer caracteres específicos que Abbyy desconocía, y no la de interpretarlos correctamente en una nueva imagen con pequeñas variaciones de otros caracteres.

Entrenar con caracteres muy deformados e ilegibles puede causar un reconocimiento de peor calidad. Por ejemplo, en la Imagen 4.2 se tiene la palabra "Directorio" , por el contexto se entiende que la cuarta letra es una “e”, aunque visualmente sea similar a una “o”. Como Abbyy utiliza un entrenamiento por carácter, especificando que el segundo carácter es una “e” puede llegar a empeorar el reconocimiento correcto de otras “o” en el texto.

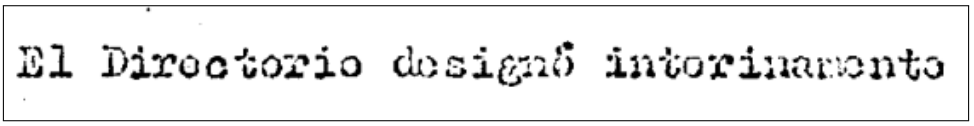


Imagen 4.2: Ejemplo de frase para entrenar Abbyy

4.1.3. Readiris 17

Readiris [24] es una herramienta creada por la empresa belga I.R.I.S enfocada en la creación de software y productos de escaneado y conversión de documentos. Esta herramienta que reconoce hasta 138 idiomas diferentes permite digitalizar documentos convirtiéndolos en diversos formatos como Word, Excel, Pdf, Html o Txt, incluso convertirlos a audio. Posee muchas capacidades para la manipulación de los documentos ya digitalizados como modificarlos, combinarlos, editarlos o anotarlos, principalmente en formato PDF. Readiris presenta tres versiones o licencias, pero también brinda la posibilidad de una prueba gratis.

Una vez utilizada la herramienta, visto los resultados y sumado a la descripción brindada del producto en su web, se llegó a la conclusión que Readiris está más enfocado en la manipulación de documentos digitales que en la función de OCR. Si bien sus resultados no eran malos, se le dificultaba mucho reconocer documentos de mala calidad.

Entrenamiento

Readiris ofrece la posibilidad de realizar un entrenamiento, pero debido a un error del mismo programa, no se logró realizarlo, cuando se intentaba entrenar en la herramienta ocurría un error. Por esta razón se experimentó con el OCR sin entrenar.

4.2. Comparación de herramientas

En esta sección se presentan y comparan los resultados obtenidos de probar los tres OCRs, Tesseract 4, Abbyy FineReader 14 y Readiris 17, para un conjunto de imágenes escaneadas de microfilm de media y mala calidad de documentos de la dictadura uruguaya

La calidad del reconocimiento del documento por parte de los OCRs se compara con las transcripciones manuales de los documentos realizadas por el equipo. Se toma en cuenta que cuanto más similar sea la salida a su respectiva transcripción manual, mejor es el funcionamiento del OCR.

Como Abbyy fue el único OCR con el cual se pudo entrenar y obtener resultados positivos, se incluyeron distintas versiones del mismo las cuales se diferencian por la cantidad de imágenes con las que se las entrenó. Por lo tanto, se consideró ABBYY entrenado con una, dos y tres imágenes mientras que Tesseract y Readiris fueron utilizados con el lenguaje por defecto para español sin entrenar.

Para medir resultados se utilizó la función ratio de la librería difflib de Python, explicada en el capítulo 2.4. Recordamos que cuanto más cercano a uno es la medida retornada por la función, más similares son los textos comparados, mientras que cuanto menor sea el valor indica que la diferencia es mayor.

Resultados

En la Tabla 4.1 se muestran los resultados obtenidos de comparar utilizando SequenceMatcher las salidas de cada OCR con su respectiva transcripción manual.

Las imágenes ent_1, ent_2 y ent_3 se usaron para entrenar Abbyy. El modelo Abbyy_1 se entrenó solo con ent_1, mientras que Abbyy_1_2 fue con ent_1 y ent_2, lo mismo para Abbyy_1_2_3 el cual utilizó ent_1, ent_2 y ent_3. En la primera sección de la Tabla 4.1 se comparan resultados con las mismas imágenes que se entrenó Abbyy. El grupo de imágenes correspondientes a las filas que sigue debajo de la línea horizontal en la Tabla 4.1 (example_i) no fue utilizado para entrenar ningún modelo, solo para medir resultados.

Imagen	Abbyy	Abbyy_1	Abbyy_1_2	Abbyy_1_2_3	Tesseract	Readiris
ent_1	0.8760	0.8768	**0.8775**	0.8722	0.6579	0.4794
ent_2	**0.7486**	0.6884	0.7114	0.6963	0.6502	0.0716
ent_3	0.0435	0.0648	0.0537	**0.2736**	0.1207	0.0102
example_1	**0.6686**	0.6529	0.6560	0.6614	0.6267	0.0181
example_2	**0.3893**	0.1274	0.2083	0.1243	0.1053	0.0131
example_3	0.0236	0.0542	0.0534	0.0647	**0.1947**	0.0285
example_4	0.3000	0.3245	**0.3667**	0.1816	0.2356	0.0122
example_5	0.7898	**0.8243**	0.7730	0.8206	0.5707	0.0336
example_6	0.1539	0.2706	**0.3061**	0.2961	0.0234	0.0086
example_7	0.4290	0.4121	**0.4510**	0.3278	0.3482	0.0115
example_8	0.2945	0.1396	**0.3737**	0.3377	0.2981	0.0039

Tabla 4.1: Tabla de comparación de OCRs

Abbyy y sus distintas versiones de entrenamiento resultaron vencedores por sobre Tesseract y Readiris. Este último tuvo una muy mala performance. La diferencia aquí vista coincide con lo sensación que nos dio ver las salidas a simple vista. Además de los resultados mostrados en la Tabla 4.1, observamos que Abbyy, en comparación con Tesseract, reconoce tanto las manchas de los bordes como una

posible inclinación en la imagen y lo corrige, por lo que esto genera usualmente una salida más similar al texto real ya que no convierte como texto muchas de las manchas contenidas en la hoja.

En la Imagen 4.3 se muestra una gráfica con el promedio de cada OCR evaluado sobre todos los ejemplos vistos, donde se aprecia la tendencia de Abbyy sobre Tesseract, y bastante más atrás está Readiris.

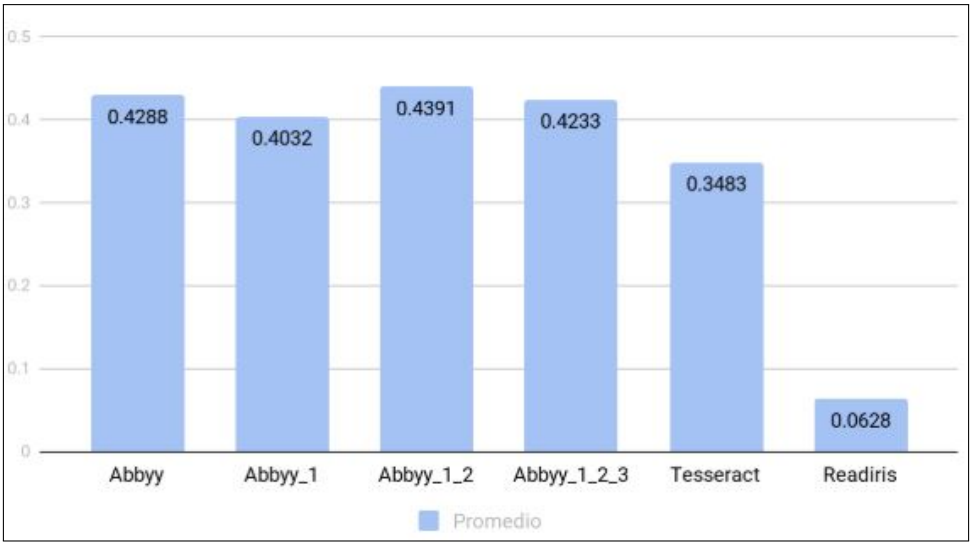
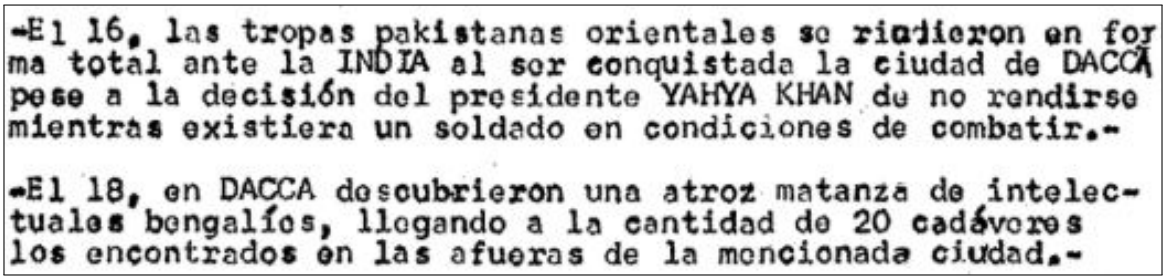


Imagen 4.3: Gráfica de comparación de OCRs

En la Imagen 4.4 se muestra una porción del documento utilizado para el entrenamiento de Abbyy (ent_1).


-El 16, las tropas pakistanas orientales se rindieron en forma total ante la INDIA al ser conquistada la ciudad de DACCA pese a la decisión del presidente YAHYA KHAN de no rendirse mientras existiera un soldado en condiciones de combatir.-

-El 18, en DACCA descubrieron una atroz matanza de intelectuales bengalíes, llegando a la cantidad de 20 cadáveres los encontrados en las afueras de la mencionada ciudad.-

Imagen 4.4: Recorte documento original ent_1

La imagen utilizada se considera de calidad media dado que como se aprecia en el recorte del documento, algunas letras se presentan difusas y pueden ser confundidas por otras letras por los OCRs. En las imágenes Imagen 4.5, Imagen 4.6, Imagen 4.7 e Imagen 4.8 se expone el texto reconocido por los distintos OCRs para el documento mostrado anteriormente.

Se puede apreciar que los resultados obtenidos por los OCRs para el recorte anterior son consistentes con los puntajes presentados en la Tabla 4.1, en promedio es Abbyy quien cometió la menor cantidad de errores.

```
"-El 16, lns tropas pokistanas orientales so rioiieron en for "
"ma total ante la INDIA al sor eonquistada la ciudad de DAC~ "
"pese a la décisi6n del presidente YAtNA KHAN_du no rendirse "
"mientras existiera un sold~do en condiciones de combatir.- "


"•El 18, en DACf,A doscu~rieron una atrot m~tanza de intelec- "
"tuales bcngalios, llegando a 1~ cantidad da 20 cad,vores "


"los encontxadc s en las afueras de la mencionada clt.tdlid ••• "
```

Imagen 4.5: Texto reconocido por Readiris

```
«+E]l 16, las tropas akistanas orientales sc riediezon en for
ma total ante la INDÍA al sor eonquistada la ciudad de DACCÁ
pese a la decisión del presidente YAHYA KHAN de no rendirse

mientras existiera un soldado en condiciones de combatirea-

«El 18, en DACCA descubrieron una atroz matanza de intelec-
tuales bengalícs, llegando a la cantidad de 20 cadávores
los encontrados en las afueras de la mencionada ciudada=
```

Imagen 4.6: Texto reconocido por Tesseract

```
*El 16, las tropas pakistanas orientales se rindieron en for
ma total ante la INDIA al ser conquistada la ciudad de DACdt
pese a la decisión del presidente YAHYA KHAN de no rendirse
mientras existiera un soldado en condiciones de combatir.-

*El 19, en DACCA descubrieron una atroz matanza de intelec-
tuales bengalíos, llegando a la cantidad de 20 cadáveres
los encontrados en las afueras de la mencionada ciudad.-
```

Imagen 4.7: Texto reconocido con Abbyy por defecto

```
-El 16, las tropas pakistanas orientales se rindieron en for
ma total ante la INDIA al sor conquistada la ciudad de DACCA
pese a la decisión del presidente YAHYA KHAN de no rendirse
mientras existiera un soldado en condiciones de combatir.-

-El 18, en DACCA descubrieron una atroz matanza de intelec-
tualee bengalíos, llegando a la cantidad de 20 cadáveres
los encontrados en las afueras de la mencionada ciudad*-
```

Imagen 4.8: Texto reconocido por Abbyy_1_2_3

La Imagen 4.9 muestra una porción de texto del ejemplo de entrenamiento 3 (ent_3), que se considera de mala calidad dado que a simple vista no se logra entender todo el texto y es difícil hacerlo incluso en su contexto.

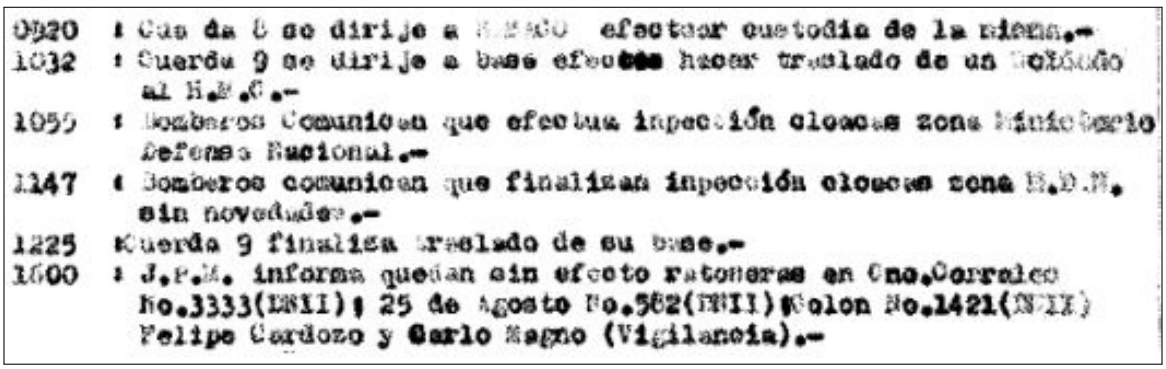


Imagen 4.9: Recorte documento original ent_3

Tanto Readiris, Tesseract y Abbyy sin entrenar al procesar el recorte de la imagen ent_3, generaron como salida alguna palabra legible y muchos caracteres sin sentido. Solo Abbyy_1_2_3 (entrenado con ent_1, ent_2 y ent_3) reconoció trigramas o frases correctamente, aunque también reconoció muchos caracteres sin sentido. Este comportamiento descrito se aprecia en los resultados de los algoritmos en la Tabla 4.1, donde la similitud está en el orden del segundo decimal excepto para Abbyy_1_2_3 el cual fue entrenado con la misma imagen.

Luego de esta evaluación, se puede concluir que Abbyy resulta la mejor opción para el primer paso de convertir las imágenes a texto, siendo superior en diez de los once ejemplos vistos por un amplio margen sobre los demás OCRs.

Enfocándose ahora en las distintas versiones de Abbyy resultado del entrenamiento, es difícil inferir a partir del reducido número de experimentos realizados (se lo puede llamar reducido si se lo compara con la cantidad total de imágenes) si el hecho de agregar parejas imagen/carácter a los modelos generará un cambio significativo a la hora de procesar el resto de las imágenes o no, sin olvidar además que existe un límite de 1000 parejas para cada modelo. La existencia de dicho límite acarrea el problema de decidir cuáles caracteres integrarán el selecto grupo y cuáles no, teniendo como fuente miles de documentos, buscando siempre el objetivo de generar la salida óptima. Se considera que atacar de llano el problema planteado anteriormente sería sumamente complejo y los experimentos para decidir con criterio conllevarían una cantidad enorme de tiempo y no aporta grandes cambios, es por esta razón que se decide utilizar Abbyy sin entrenar como OCR definitivo para reconocer los documentos.

Capítulo 5

Solución basada en modelos de lenguaje

Este tipo de enfoques se basan en la utilización de probabilidades de ocurrencia de n-gramas. Dado un error y las posibles correcciones de este, se busca reemplazarlo por la opción cuya probabilidad de ser la acertada sea más alta.

A grandes rasgos este proceso implica detectar errores, obtener posibles candidatos para corregirlos y tener mecanismos para optar por el candidato más adecuado. Todos estos pasos hacen uso de vocabularios.

5.1. Vocabularios o diccionarios

Un vocabulario o diccionario en el ámbito de procesamiento de lenguaje natural se refiere a un conjunto de palabras que forman el universo de palabras correctas. En nuestro caso y a lo largo del informe se prefirió el uso de la palabra vocabulario para evitar confusión con la estructura de programación diccionario (o mapa). En este proyecto se definió considerar toda palabra contenida en los documentos que se encuentre presente en el vocabulario como una palabra correcta. De esta forma, la correctitud de una palabra será evaluada considerando únicamente la palabra y no se considerará el contexto en el que se encuentre. Un detalle a tener en cuenta es que un error del OCR puede pasar desapercibido si se reconoce una palabra en un documento como otra que se encuentre en el vocabulario, si sucede esto, la palabra reconocida erróneamente por el OCR será considerada correcta igualmente. Por otro lado, para corregir una palabra errónea se cuenta con posibles candidatos, estos posibles candidatos para corregir una palabra deberán pertenecer y ser buscados dentro de este universo.

Para construir nuestro vocabulario primero se procedió a seleccionar un vocabulario base, el cual se incrementó, limpió y mejoró en el correr del proyecto.

En la búsqueda del vocabulario base se analizaron varios vocabularios en español. Por la cantidad de palabras y la calidad de las mismas se decidió comenzar trabajando con el vocabulario de FreeLing [25]. FreeLing es un kit de herramientas de código abierto de procesamiento de lenguaje natural que brinda un vocabulario del idioma español.

El siguiente paso constó del enriquecimiento de la base con la que se contaba con las abreviaturas militares proporcionadas por un proyecto de la Facultad de Información y Comunicación de la UDELAR. Este proyecto busca listar todas las siglas militares utilizadas en documentos de la época de la dictadura

Luego al vocabulario se le agregó el vocabulario RLA-ES [26] (Recursos Lingüísticos Abiertos del Español). Este es un proyecto colaborativo para el desarrollo de herramientas de asistencia a la escritura (corrector ortográfico, tesauro, y en un futuro corrector gramatical) para la lengua española en todas sus variantes regionales incluidas la uruguaya. Sus expectativas son las de generar el mejor corrector ortográfico para la lengua española.

Por último, se realizó un proceso de agregado y filtrado de palabras en base a resultados prácticos. Analizando los documentos corregidos se observaron errores frecuentes causados por palabras poco comunes o que dentro del contexto temporal no tendrían sentido. Por ejemplo, no se detectaban como errores palabras que desconocíamos o que entendíamos no debían estar en el documento. Algunos ejemplos de este tipo de palabras removidas fueron “os” (palabra utilizada únicamente en el español hablado en España) o “focha” (tipo de ave acuática). Estas dos palabras de ser reconocidas por el OCR no deberían pertenecer al vocabulario para no ser parte de la solución, es poco probable que un documento las contenga por lo tanto la solución no debería contenerlas tampoco. Podrían causar errores al querer corregir palabras similares como “los” o “fecha”. En total se eliminaron alrededor de veinte palabras y se agregaron aproximadamente treinta.

Vale la pena comentar que el vocabulario construido contiene formas flexivas (verbos conjugados, palabras en plural, etc.). Normalmente, un vocabulario solo contiene lemas (verbos en infinitivo, nombres en singular, etc.).

5.2. Preprocesamiento

Para poder aplicar un procesamiento de corrección basado en modelo de lenguaje, antes se tiene que realizar un preprocesamiento de los datos, este será el encargado de limpiar el texto de posibles condiciones que afecten negativamente el desempeño del corrector.

Limpieza de caracteres

Por medio de revisión manual, a lo largo del proyecto fuimos identificando ciertas equivocaciones recurrentes cometidas por el OCR al leer determinados caracteres, así como patrones utilizados en los documentos por quienes los escribieron que generaban dificultades a la hora de corregir. Es así que reemplazamos y eliminamos distintos tipos de caracteres utilizando expresiones regulares.

1. Llevar los distintos tipos de comillas a una forma común o estándar, reemplazándolas todas por comillas dobles, esto es debido a que los OCRs cuando procesan imágenes de mala calidad suelen confundir unas por otras. Unificándolas se puede evitar este tipo de errores. Las comillas a reemplazar son ‘, ', ′ por comillas dobles verticales ".
2. En los informes y documentos se utilizaba la expresión “.-” para terminar una oración, una suerte de punto final. Esta expresión no aporta mayor información que el punto final “.” al texto y al procesamiento, e incluso puede inducir errores ya que es tokenizada como dos símbolos por separado los cuales tienen una semántica distinta en el lenguaje. Por lo tanto se decide reemplazarla por un “.” únicamente.
3. Un error común detectado generado por el OCR es el de confundir los guiones al final de línea “-” por dos asteriscos (“**”), por lo tanto durante el preprocesamiento si se detecta una palabra seguida de dos asteriscos y un salto de línea, se procede a reemplazar los asteriscos por un guion. De esta forma durante la etapa del procesamiento se puede corregir o unir las palabras separadas por un guion y un salto de línea.

4. Uno de los caracteres que suele reconocer erróneamente el OCR es el de *ampersand* o "&". Debido a la calidad de algunas imágenes los caracteres "a", "e" u otros caracteres sobre los cuales desafortunadamente encontramos manchas o distorsiones por la tinta pueden ocasionar que el OCR termine identificando un carácter dentro de una palabra como "&". Esto genera que el OCR reconozca palabras como "p&trulla". Al tokenizar el texto durante el procesamiento se considera al & como un token separado, es por ello, que de la tokenización de p&trulla se obtiene ("p","&","trulla") lo que hace imposible una corrección luego. Por lo tanto, se decidió reemplazar el carácter "&" por un carácter que no se encuentre dentro de las salidas generadas por el OCR y que no sea considerado un separador de tokens, este nuevo carácter es el "½".

   La palabra "p&trulla" se convierte en "p½trulla" y al tokenizarse se tiene una única palabra.

   En resumen, en la etapa de preprocesamiento se realizan seis tipos de remplazo de caracteres.

carácter original	Remplazo
‘	"
'	"
′	"
.-	.
**	-
&	½

Tabla 5.1: Tabla de remplazos.

Palabras cortadas

Otro error que suele suceder en los procesamientos de documentos hechos en máquinas de escribir es la aparición de espacios entre letras de una misma palabra. Este error muchas veces es causado debido a la calidad de la imagen o a la calidad de escritura y genera que una sola palabra se divida en dos o más palabras. Un ejemplo de esto podría ser que de la palabra *empezaron* se reconozca como *empe zaron* o *empe zar on*.

Generalmente las subpartes de una palabra que fue dividida son reconocidas como errores por no pertenecer al vocabulario, en el ejemplo de *empe zaron* y *empe zar on* se reconoce cada subparte como error, este hecho desencadenará que en la etapa del procesamiento se quiera corregir cada una por separado. Al intentar corregirlas, se obtendrán dos o tres palabras, lo que difiere del valor correcto que sería tener únicamente la palabra *empezaron*.

Por lo tanto, se quiere unir las palabras que fueron divididas erróneamente, para eso se realiza un preprocesamiento en el cual se tokeniza cada documento y se recorren los tokens en busca de errores. Cuando una palabra es detectada como error (no se encuentra en el vocabulario) se considera candidata a ser una palabra dividida por espacios, supongamos el token $tok1$. A continuación se prueba si el siguiente token $tok2$ es considerado error. Si $tok2$ es error, se verifica si la palabra $tok1tok2$ es considerada palabra válida, de serlo entonces se entiende que $tok1tok2$ era la palabra original y es reemplazado en el texto. Si $tok1tok2$ sigue siendo considerada error se repite el paso anterior y se prueba con $tok3$, si $tok3$ es un error y $tok1tok2tok3$ es una palabra válida se remplaza en el texto, de lo contrario, se asume que $tok1$ no pertenece a una palabra dividida y se continúa el recorrido de los tokens del documento. De esta forma se logra unir palabras separadas hasta en tres partes en donde todas las partes son error.

Este procesamiento también puede unir palabras que poseen un salto de línea entre medio y no posean un guion al final de la línea superior. Dicho comportamiento se observó varias veces en los documentos.

5.3. Procesamiento

Para aplicar la corrección en los documentos, hay que asumir algunos detalles de estructura y sintaxis y decidir en base a los mismos. Para eso, se realizó un experimento para evaluar qué decisiones tomar a la hora de corregir errores. En esta sección se muestra el experimento realizado y se explican los detalles de los métodos utilizados para corregir palabras a través de distintos modelos de lenguaje.

5.3.1. Ajustes de parámetros

A la hora de procesar los textos para corregir errores, hay que asumir o definir algunos parámetros de decisión de forma que se maximice la cantidad de errores detectados y errores corregidos correctamente, y se minimice la cantidad de errores no corregidos, errores corregidos incorrectamente y/o errores no detectados. En definitiva, se busca que se puedan detectar y corregir la mayor cantidad de errores reales posibles y evitar la detección de errores falsos. Para esto, se pueden definir infinitas posibilidades o alternativas de procesamiento, por ejemplo: cuándo una palabra contiene error, el sentido o dirección de lectura de los textos (de izquierda a derecha o de derecha a izquierda), el tamaño del contexto utilizado para la corrección de un error, etc.
A priori no podemos saber qué configuración es la mejor. Si bien podemos hacer especulaciones basándonos en resultados de trabajos similares la palabra final estará dada por la performance de las distintas estrategias en la práctica. Es por esta razón que la solución se implementó de tal manera de poder cambiar y probar fácilmente algunos parámetros para poder realizar pruebas con todas las combinaciones posibles, y luego, evaluando los resultados, tomar la configuración más óptima para el problema presentado.

Entre estos parámetros se encuentran:

- **vocabulary**: usada para definir qué vocabulario usar. El vocabulario es la lista de palabras consideradas como correctas. Entonces, si una palabra determinada no se encuentra en el vocabulario, se toma como errónea.
  Se definieron dos vocabularios:
  - *elmo_vocabulary*: vocabulario generado por un colega para su proyecto de grado de entrenamiento de ELMo para español [27].
  - *vocabulary1.5*: vocabulario generado a lo largo del proyecto, a partir de la combinación de los vocabularios de FreeLing [28], RLA-ES [29] de Uruguay y Argentina y modificaciones manuales.
- **correct_upper_case**: variable booleana utilizada para asumir (o no) que una palabra escrita completamente en mayúscula no es un error (ignorando el vocabulario). El objetivo es evitar falsos positivos con siglas.
- **correct_upper_case_first_letter**: variable booleana utilizada para asumir (o no) que una palabra, excepto si es de inicio de oración, escrita con la primera letra en mayúscula no es un error (ignorando el vocabulario). El objetivo es evitar falsos positivos con nombres propios, apellidos, países, etc. que podrían no pertenecer al vocabulario.

- **context_direction**: define la dirección del contexto utilizado al trabajar con el modelo de lenguaje.
  Pasaremos a mostrar los valores que puede tomar este parámetro junto con ejemplos del resultado de cada elección. Consideremos un contexto de largo dos para facilitar la explicación.

  - *previous*: el contexto utilizado son las dos palabras anteriores al error.
  - *middle*: el contexto utilizado es la palabra anterior y la siguiente al error.
  - *forward*: el contexto utilizado son las dos palabras posteriores al error.
  - *all*: se usan los tres contextos.

  Por ejemplo, dada la frase con error: "Se detuvo a senor en actitud sospechosa".
  Los contextos serían:

  Previous: "detuvo a".

  Middle: "a" y "en".

  Forward: "en actitud".

- **process_split_word_by_newline**: indica cómo trabajar con palabras divididas por un guion y salto de línea. El objetivo es encontrar la mejor forma de lidiar con esta situación particular. ¿Resulta conveniente unir los segmentos en cuestión y trabajar como si nunca hubieran estado separados? ¿Es mejor mantenerlos divididos considerando a cada uno como una palabra correcta? A través de las pruebas realizadas con los distintos valores que puede tomar este parámetro buscamos dar respuestas a estas interrogantes. A continuación, listamos las posibles opciones:

  - *not_procces*: se saltea el procesamiento de estos casos particulares.
  - *join*: se junta y se mantiene la palabra entera para el resto del procesamiento.
  - *split*: se mantiene la palabra separada para el resto del procesamiento pero se marca para que no sea considerada como error posteriormente.

- **correct_split_word**: utilizada para corregir palabras que se encuentran completas en los documentos pero el OCR reconoció como varias palabras de menor largo divididas por espacios en blanco. Trabaja suponiendo hasta un máximo de tres partes. Para este parámetro hay tres posibles configuraciones:

  - *not_process*: se desactiva este procesamiento.
  - *same_line*: corrige divisiones solo si pertenecen a la misma línea (sin verificar si la palabra continúa en el siguiente renglón).
  - *any_line*: corrige palabras divididas incluso si es por un salto de línea.

- **language_model**: indica qué modelo de lenguaje utilizar entre los disponibles.
  Se usaron dos posibles modelos de lenguaje explicados en la sección 5.3.3:

  - *google*: Modelo de lenguaje basado en el conjunto de n-gramas publicado por Google [30].
  - *1_billion*: Modelo de lenguaje construido por nosotros a partir del corpus en español *Spanish Billion Word Corpus and Embeddings* [31].

Comparación de configuraciones

Teniendo en cuenta los parámetros y sus posibles valores, se cuenta con 576 ($2 \times 2 \times 2 \times 4 \times 3 \times 3 \times 2 = 576$) configuraciones posibles, sobre las cuales se quiere obtener la óptima para procesar todos los documentos de nuestro universo. La comparación de los resultados se realiza utilizando la medida obtenida a partir de SequenceMatcher explicada en el Sección 2.4 para medir similitud entre textos. Comparamos qué tan similares son el texto corregido y sin corregir respecto del *ground truth*. Las configuraciones que obtengan un mayor nivel de similitud para los textos corregidos son consideradas mejores.

Para la evaluación, se realizó la corrección de 100 documentos elegidos a mano (priorizando los documentos más legibles, por ser de los que se obtienen correcciones más interesantes) para cada una de las configuraciones posibles. Estos 100 documentos, constituyen casi un 10 % del total de documentos disponibles junto con su transcripción obtenida mediante LUISA (*ground truth*) hasta ese momento, por lo tanto se considera una cantidad razonable de ejemplos para generalizar el comportamiento de cada configuración para el resto de los documentos del universo.

Para comparar los resultados se compararon las siguientes métricas:

1. **Cantidad de victorias**: para cada documento, se considera victoriosa aquella configuración que obtiene el mayor puntaje de similitud, por lo tanto la configuración con mayor cantidad de victorias es considerada mejor por esta métrica. Para un mismo documento puede haber más de una configuración victoriosa.
2. **Cantidad de documentos con incremento en puntaje**: para cada configuración, cuantos documentos luego de ser procesados incrementaron o redujeron el puntaje de similitud. La configuración que maximiza la cantidad de documentos que incrementaron el puntaje y minimice la cantidad de documentos que redujeron el puntaje es considerada la mejor.
3. **Promedio de similitud**: el promedio de puntaje de similitud de cada configuración en los 100 documentos. Cuanto mayor es el número, mejor es la corrección en general de todos los documentos, por lo tanto la configuración con mayor puntaje es evaluada como mejor según esta métrica.
4. **Percentil 75**: indica el puntaje límite para el cual el 75 % de las pruebas se encuentra por debajo de este. Por lo tanto, se considera que una configuración es mejor si tiene mayor percentil.

Para entender las diferencias entre los puntajes de similitud de las medidas 3 y 4 y saber aproximadamente qué tanto varían los puntajes de similitud al corregir un número de errores, se presenta el fragmento de un documento de ejemplo con una posible corrección y sus respectivos puntajes.

La Tabla 5.2 muestra un fragmento de un documento con errores (salida OCR) y el texto obtenido de LUISA (salida LUISA) utilizado como la mejor transcripción del documento que se quiere obtener. El puntaje de similitud entre estos dos textos es de 0.92327.
Con las correcciones resaltadas en verde en la Tabla 5.3 el puntaje de similitud es de 0.95178, una diferencia de 0.02851 puntos más que el texto antes de corregir. Los errores corregidos en este caso son todas las palabras que no existen en el idioma español en el texto antes de corregir, sin contemplar las palabras que empiezan en mayúscula y que no son palabras de inicio de oración, esto último debido a que más adelante se verá que las configuraciones que siguieron este enfoque obtuvieron mejores resultados.
La corrección que se hizo es manual y solo a efectos de poder ver la diferencia de puntajes obtenida. Las correcciones en verde de la Tabla 5.3 suponen una corrección relativamente buena, con lo que se

puede asumir que una mejora de alrededor de 0.02 en el puntaje se trata de una corrección interesante. Una mejora chica en la corrección (corrigiendo solo una letra) y su relativo puntaje de similitud se puede ver en la Tabla 5.4, con una diferencia de 0.00252 en el puntaje, por lo tanto se asume que un cambio en el puntaje de este margen no supone un cambio significativo.

Salida OCR	Salida LUISA
Personal da ósta DNII afectado al aeropuerto Nacional da Carrasco, procedió, sn sl^ día da ayer, a registrar la entrada al país dsl ciudadano Dr.Jusn Pablo TERRA, que'í ingresaba sí mismo proveniente ds Río da Janeiro en el vuelo 405 da la Coqpañíz [Cruzeiro do Bul de la hora 15.45*. Registrado minuciosamente su equipaje, y tata- t¡ bl&i su persona, dicha inspección arrojó resultadas negativos.- í	Personal de esta DNII afectado al Aeropuerto Nacional de Carrasco, procedió, en el día de ayer, a registrar la entrada al país del ciudadano Dr. Juan Pablo TERRA, que Ingresaba al mismo proveniente de Río de Janeiro en el vuelo 405 de la Compañía Cruzeiro do Sul de la hora 15.45\'. Registrado minuciosamente su equipaje, y tambien su persona, dicha inspección arrojó resultados negativos.-

Puntaje: 0.92327

Tabla 5.2: Fragmento de documento sin corregir, salida obtenida a partir de LUISA y puntaje de similitud

Salida OCR corregida	Salida LUISA
Personal da esta DNII afectado al aeropuerto Nacional da Carrasco, procedió, en el día da ayer, a registrar la entrada al país del ciudadano Dr.Jusn Pablo TERRA, que ingresaba sí mismo proveniente de Río da Janeiro en el vuelo 405 da la Coqpañíz [Cruzeiro do Bul de la hora 15.45*. Registrado minuciosamente su equipaje, y tambien su persona, dicha inspección arrojó resultadas negativos.- í	Personal de esta DNII afectado al Aeropuerto Nacional de Carrasco, procedió, en el día de ayer, a registrar la entrada al país del ciudadano Dr. Juan Pablo TERRA, que Ingresaba al mismo proveniente de Río de Janeiro en el vuelo 405 de la Compañía Cruzeiro do Sul de la hora 15.45\'. Registrado minuciosamente su equipaje, y tambien su persona, dicha inspección arrojó resultados negativos.-

Puntaje: 0.95178

Tabla 5.3: Fragmento de documento corregido, salida obtenida a partir de LUISA y puntaje de similitud

Salida OCR	Salida LUISA
Personal da esta DNII afectado al aeropuerto Nacional da Carrasco, procedió, sn sl^ día da ayer, a registrar la entrada al país dsl ciudadano Dr.Jusn Pablo TERRA, que'í ingresaba sí mismo proveniente ds Río da Janeiro en el vuelo 405 da la Coqpañíz [Cruzeiro do Bul de la hora 15.45*. Registrado minuciosamente su equipaje, y tata- t¡ bl&i su persona, dicha inspección arrojó resultadas negativos.- í	Personal de esta DNII afectado al Aeropuerto Nacional de Carrasco, procedió, en el día de ayer, a registrar la entrada al país del ciudadano Dr. Juan Pablo TERRA, que Ingresaba al mismo proveniente de Río de Janeiro en el vuelo 405 de la Compañía Cruzeiro do Sul de la hora 15.45\'. Registrado minuciosamente su equipaje, y tambien su persona, dicha inspección arrojó resultados negativos.-

Puntaje: 0.92579

Tabla 5.4: Fragmento de documento con una sola corrección, salida obtenida a partir de LUISA y puntaje de similitud

A continuación se muestran los primeros tres puestos (ordenadas de mejor a peor configuración) para cada una de las métricas, como también en algunos casos, configuraciones que se encuentran por debajo del puesto 3, con el objetivo de compararlas, debido a que esas configuraciones son ganadoras en al menos una de las otras métricas.

En algunos casos las métricas tienen un conjunto de configuraciones que comparten el mismo puesto. Por esta razón se indica, para cada métrica, la cantidad total de puestos, con el objetivo de identificar si un puesto en el que se encuentra una configuración es bueno o no en relación a todas las demás configuraciones (no es lo mismo que una configuración se encuentre en el puesto 7 cuando existen 10 puestos, que cuando existen 500).

1. Cantidad de victorias

Datos en los resultados a tener en cuenta:

- Total de puestos: 15.
- Para muchos documentos, el puntaje de dos o más configuraciones distintas fue el mismo, es por esto que el total de victorias sumadas supera el valor 100.

Como ya se dijo anteriormente, esta métrica indica la cantidad de veces que una determinada configuración obtuvo el mayor puntaje para un documento en particular.

En la Tabla 10.2 del anexo se muestran los resultados completos de la prueba para esta medida, la Tabla 5.5 muestra las nueve configuraciones (puestos 1, 2 y 3) con mayor cantidad de victorias y en el puesto siete, la **config67** con la mitad de victorias que la mejor para esta medida. Analizando los datos de la tabla, la configuración que obtuvo mejor puntaje de similitud la mayor cantidad de veces fue **config20**, en particular, en 16 de los 100 documentos. Por otro lado, la **config67** (mejor configuración para las medidas 3 y 4) obtuvo el mejor puntaje para la mitad de documentos que la mejor configuración en esta medida.

Puesto	Configuración	Cantidad victorias
1	**config20**	**16**
2	**config19**	**14**
3	**config21**	**12**
3	config298	12
3	config299	12
3	config300	12
3	config310	12
3	config311	12
3	config312	12
	...	
7	config67	8

Tabla 5.5: Top 3 configuraciones con mayor cantidad de victorias.

2. Cantidad de documentos con incremento en puntaje

Datos en los resultados a tener en cuenta:

- Total de puestos: 65.
- Las peores configuraciones incrementaron el puntaje solamente en 11 documentos, y obtuvieron un decremento del mismo en 89 documentos.

La Tabla 10.4 del anexo muestra el resultado de esta medida, que se obtiene de comparar el puntaje de similitud obtenido entre el documento transcrito original (salida de Abbyy) y la salida de LUISA, contra el mismo documento luego de ser corregido y la salida de LUISA, con lo que se puede asumir que si el puntaje luego del procesamiento es mayor para más cantidad de documentos, la configuración será mejor en general.
En la Tabla 5.6 se observan los primeros 3 puestos y en el puesto 4 se encuentra la **config67**. En este caso las configuraciones ganadoras son las mismas tres que para la medida anterior, pero empatando en el resultado. En comparación con la **config67**, estas tres configuraciones obtuvieron una mejora de puntaje en tres documentos más, una cantidad relativamente alta considerando que el total de documentos en la prueba es cien.

Puesto	Configuración	Cantidad incrementos	Cantidad decrementos
1	**config19**	**82**	**18**
1	**config20**	**82**	**18**
1	**config21**	**82**	**18**
2	config13	81	19
2	config14	81	19
2	config15	81	19
2	config301	81	19
2	config302	81	19
2	config303	81	19
2	config307	81	19
2	config308	81	19
2	config68	81	19
2	config69	81	19
2	config7	81	19
2	config8	81	19
2	config9	81	19
3	config9	80	20
3	config10	80	20
3	config11	80	20
3	config12	80	20
3	config295	80	20
3	config297	80	20
3	config309	80	20
3	config349	80	20
3	config63	80	20
4	**config67**	**79**	**21**

Tabla 5.6: Top 3 configuraciones con mayor cantidad de documentos con incremento de puntaje.

3. Promedio de similitud

Datos en los resultados a tener en cuenta:

- Total de puestos: 576 (misma cantidad que configuraciones).
- El promedio de puntaje para los 100 documentos sin corregir es 0.76079.
- Hay un total de 404 configuraciones con un promedio menor al del promedio de los documentos sin corregir, por lo tanto 172 configuraciones con promedio mayor.

A grandes rasgos, esta medida muestra para cada configuración, qué tanto se corrigieron los documentos en conjunto, sin observar cada uno por separado. La Tabla 5.7 muestra los primeros 3 puestos y también en el puesto 22 y 27 las **config19** y **config21** respectivamente, dos de las configuraciones ganadoras de las medidas anteriores.
Aunque la **config21** se encuentra muchos puestos más abajo que la **config67**, la diferencia entre puntajes es muy pequeña (0.00153), y más aun considerando la diferencia entre estas y el puntaje del promedio calculado con los documentos sin corregir (0.00606 y 0.00453 respectivamente), por lo tanto es razonable considerar que para esta medida cualquiera de las configuraciones es similar y por lo tanto tiene menos peso a la hora de decidir qué configuración es mejor.
Los resultados completos de esta medida se pueden ver en la Tabla 10.3 del anexo.

Puesto	Configuración	Promedio
1	**config67**	**0.76685**
2	**config20**	**0.76683**
3	config61	0.76671
	...	
22	**config19**	**0.76539**
	...	
27	**config21**	**0.76532**

Tabla 5.7: Configuraciones con mayor promedio de puntaje respecto a los 100 documentos.

4. Percentil 75

Datos en los resultados a tener en cuenta:

- Total de puestos: 338.
- El percentil 75 para los 100 documentos sin corregir es 0.86623.
- Hay 404 configuraciones con un percentil 75 menor al del valor de la medida de los documentos sin corregir, por lo tanto 172 configuraciones con percentil 75 mayor.

Un percentil es una medida que representa, una vez ordenados los datos de menor a mayor, el valor de la variable por debajo del cual se encuentra un porcentaje dado de observaciones en un grupo. Por ejemplo, el percentil 75 de una configuración, indica el puntaje límite para el cual el 75 % de las pruebas se encuentra por debajo de este.
Los resultados completos de la prueba se ven en la Tabla 10.5 del anexo, y en la Tabla 5.8 se observan los primeros 4 puestos, siendo ganadora la **config67** al igual que en la medida 3 y los 3 restantes los ganadores de las medidas 1 y 2. De esta manera, se reafirma en forma general que estas configuraciones son las mejores. Al igual que sucede en la medida anterior, los puntajes de las configuraciones

ganadoras difieren mínimamente por lo que se asume que usar cualquiera de ellas es una buena decisión.

Puesto	Configuración	Percentil 75
1	**config67**	**0.87612**
2	**config19**	**0.87585**
3	**config20**	**0.87575**
4	**config21**	**0.87548**

Tabla 5.8: Top 4 configuraciones con mayor percentil 75 respecto a los 100 documentos.

Resultados

La Tabla 5.9 muestra el valor de cada uno de los parámetros de las cuatro configuraciones que obtuvieron mejores resultados en las pruebas (en la Tabla 10.1 del anexo se muestran todas las configuraciones). Hay cinco parámetros que comparten el mismo valor en todas las configuraciones comparadas respectivamente, esto nos da una fuerte confianza de que estos son los valores óptimos para nuestro problema. Entre estos se encuentran: usar el vocabulario generado a lo largo del proyecto, en lugar del obtenido a partir de otro proyecto; no considerar errores en palabras escritas completamente en mayúscula, ni tampoco las que empiezan con una letra en mayúscula sin ser de inicio de oración (se comprueba que corregir este tipo de palabras empeora la corrección); considerar todos los posibles contextos en la búsqueda de la corrección del error y utilizar el modelo de lenguaje de Google en lugar del nombrado "1 Billion" construido a partir del corpus *Spanish Billion Word Corpus and Embeddings* [31].

Los resultados obtenidos para los parámetros anteriores no llaman la atención, era de esperarse que el uso del vocabulario construido particularmente en el proyecto obtendría mejores resultados que uno genérico o que las palabras con mayúscula se tienen que tratar de manera especial ya que se conoce de antemano que los documentos tienen gran cantidad de nombres propios y siglas que pueden identificarse como error aunque no lo sean. La decisión de usar el modelo de lenguaje de Google o "1 Billion" no era tan clara, pero a priori se sabía que los n-gramas de Google cuentan con una cantidad base de palabras mucho mayor, una característica sumamente importante en los modelos de lenguaje. El último de estos cinco parámetros obtuvo como mejor decisión usar todos los posibles contextos de un error en la búsqueda de su corrección. Esta decisión tiene como diferencia entre las opciones disponibles, que se evalúa un conjunto más grande de n-gramas y sus probabilidades respectivas para una posible corrección. Para nuestro problema, en base a los resultados obtenidos podemos decir que esto supone una ventaja.

Los parámetros restantes son los dos utilizados para tener en cuenta particularmente a las palabras separadas por guion y salto de línea debido al final de la hoja, y las palabras separadas simplemente porque el OCR reconoció un espacio en blanco en medio erróneamente. Para el parámetro correct_split_word se encuentran todos los valores posibles en alguna configuración, lo que da a entender que cualquiera de las posibilidades no provoca un impacto notorio con respecto a las demás. En relación al parámetro process_split_word_by_newline, destaca que el posible valor *join* no aparece en ninguna de las configuraciones ganadoras, recordando que este valor junta este tipo de palabras y luego la comparación para obtener el puntaje de similitud se realiza contra el *ground truth* también procesado para juntar las palabras separadas por salto de línea. Esto último podría juntar (o no) palabras erróneamente y puntajes más bajos en consecuencia.

En conclusión tomamos como mejor configuración a la número 20. Esta resultó ganadora tanto en la métrica de cantidad de victorias como en el conteo de documentos que vieron incrementado su puntaje. Hablando del promedio de similitud, quedó en un segundo puesto con una diferencia entre el primero de apenas 0.00002, la cual a efectos prácticos es despreciable. Finalmente, también integra el podio si nos referimos al percentil 75.

Configuración	vocabulary	c_u_c	c_u_c_f_l	c_d	p_s_w_b_n	c_s_w	l_m
config19	vocabulary1.5	FALSE	FALSE	all	split	any_line	google
config20	vocabulary1.5	FALSE	FALSE	all	split	same_line	google
config21	vocabulary1.5	FALSE	FALSE	all	split	not_process	google
config67	vocabulary1.5	FALSE	FALSE	all	not_process	any_line	google

Tabla 5.9: Detalle configuraciones ganadoras.

c_u_c = correct_uper_case c_u_c_f_l = correct_upper_case_first_letter p_s_w_b_n = process_split_word_by_newline
c_d = context_direction l_m = language_model c_s_w = correct_split_word

5.3.2. Detección de errores

Como se explicó brevemente en la sección 5.1, la detección de errores se realiza a través de la pertenencia o no al vocabulario utilizado. Por ejemplo, la palabra "constituye" se considera correcta si dentro del vocabulario utilizado hay una ocurrencia de esta, de no ser así entonces se cataloga como errónea. Por este motivo es muy importante tener un vocabulario lo más completo posible.

Las palabras al inicio de oración o en títulos de los documentos pueden estar en mayúsculas mientras que en el vocabulario pueden estar solo en minúsculas, comprobar únicamente la pertenencia sin importar si hay mayúsculas podría catalogar palabras correctas como error, por lo tanto se decidió siempre transformar en minúscula las palabras antes de buscar la pertenencia al vocabulario.

Continuando con el tema de mayúsculas y minúsculas, se observó que los nombres propios, nombres de calles o siglas, conocidos por su nombre en inglés *named entities*, se escriben en mayúsculas la mayoría de las veces. Muchas de estas palabras pueden ser muy particulares, implicando que su probabilidad de estar considerados dentro del vocabulario sea baja, pero esto no quita que sean palabras correctas. El objetivo de los parámetros *correct_upper_case* y *correct_upper_case_first_letter* explicados en la sección anterior fue verificar si ignorar este tipo de palabras (considerarlas como correctas sin buscar en el vocabulario) da mejores resultados finales en la corrección.

El parámetro *correct_upper_case* tiene como objetivo evitar evaluar una palabra cuando está escrita totalmente en mayúscula, considerándola como correcta en todos los casos. Entonces por ejemplo veamos el caso de la frase "Las empresas UTE Y ANCAP": si no discriminamos ninguna palabra, se procede a evaluar todas las palabras de la frase en busca de error. En el caso de que las palabras "UTE", "ANCAP" no se encuentren en el vocabulario, se considerarán como errores y se les buscará su corrección. La salida del procesamiento podría ser "las empresas te y ancas" (suponiendo que "te" y "ancas" se encuentran en el vocabulario), debido a que "te" y "ancas" son similares a "UTE" y "ANCAP". El resultado en lugar de corregir, termina agregando más errores a la frase. En la sección previa se comprobó que se obtienen mejores resultados ignorando el chequeo de error en este tipo de palabras, por lo tanto en esta frase "UTE" y "ANCAP" no serían considerados error y el resultado sería correcto. Por otro lado, en el caso de la frase "JOSÉ BUIS FERNANDEZ", al ignorar las palabras en mayúsculas la frase se mantendrá sin cambios por tomar como palabras correctas a las tres palabras

de la frase, y se pierde la posibilidad de detectar “BUIS” como error y corregirla correctamente por “LUIS”.

La principal motivación para la creación del parámetro *correct_upper_case_first_letter* son los nombres propios y apellidos, estas palabras pueden ser especiales y no existir en el vocabulario aunque sean correctas.
Un ejemplo es el nombre “Yamandú”, que si es evaluada y se reconoce el como error, podría corregirse a algo similar como por ejemplo “amando” por su cercanía. En este caso se introduce el error, por el contrario si se evita evaluar la palabra por iniciar en mayúscula no se corregiría el nombre y se mantendría la palabra original. Como se comprueba en la sección anterior, en algunos casos se consigue mejor resultados distinguiendo esas palabras mientras que en otros no, pero en la mayoría de los casos es mejor no evaluarlas.

5.3.3. Corrección de errores

Luego de resuelta la detección de errores el siguiente paso es su corrección.

En una de las soluciones estudiadas se intentarán corregir los errores por medio de modelos de lenguaje. Estos modelos funcionan utilizando la probabilidad de ocurrencia de n-gramas, dicho de otra forma, utiliza el contexto circundante a la palabra errónea.

Para explicar el concepto anterior primero se debe entender cómo se elige una palabra para reemplazar el error. A cada palabra errónea se le asocia un conjunto de palabras correctas candidatas a reemplazar según su distancia de edición, recordemos que la distancia de edición representa la cantidad de modificaciones individuales que se le debe realizar a una palabra para formar otra.

Estas palabras candidatas se obtienen del vocabulario definido anteriormente y se priorizan según la distancia que posean a la palabra errónea. Por ejemplo, la palabra con error "ferliz" tiene distancia uno con la palabra "feliz" y distancia dos con "perdiz". Las palabras candidatas a corregir el error con distancia menor o igual a dos de "ferliz" serían "feliz", "perdiz" en ese orden.

Para obtener las palabras candidatas se utiliza generalmente algoritmos de eliminación, reemplazos, transposición e inserción. Para esta solución se utilizó el algoritmo del proyecto Symspell [32] el cual transforma las operaciones mencionadas en únicamente la operación de eliminación de caracteres. Las operaciones de remplazo e inserción son costosas y dependen de lenguaje, en el caso del lenguaje chino existen 70.000 posibles caracteres. Al ser la eliminación de caracteres poco costosa y gracias a un precálculo que se realiza, el Symspell logra ser hasta un orden de seis veces más rápido que los algoritmos estándar.

Este proyecto permite indicar el vocabulario en el cual buscar los candidatos, la cantidad de candidatos a retornar y la distancia de edición máxima a obtener.

En un inicio la distancia de edición utilizada era de dos, esto lograba que las palabras candidatas fueran similares al error detectado. Luego de estudiar los resultados se observó que muchas veces no se estaban corrigiendo palabras largas con errores y que las palabras cortas podían cambiar demasiado. Una palabra corta de tres letras por ejemplo, podría ser reemplazada por otra que únicamente coinciden en un carácter, perdiendo totalmente a la palabra original. Las palabras largas muchas veces no se corregían debido a que no se encontraban palabras similares a la palabra incorrecta en nuestro vocabulario utilizando la distancia de edición máxima, o dicho de otra manera, no se contaba con posibles palabras que pudieran reemplazar a la incorrecta para la distancia máxima de búsqueda utilizada. Es por esta razón que se decidió probar con distancias de edición variables según el largo de la palabra a corregir, incrementando la distancia máxima de búsqueda según incrementa el largo de la palabra

errónea. En la sección 10.4 del anexo se adjuntan las distintas configuraciones exploradas junto a los resultados detallados de las pruebas. Estas pruebas tuvieron como mejor opción la siguiente variación de distancia de edición.

Palabras con:

- largo ≤ 3: distancia máxima de edición de 1.
- $3 <$ largo ≤ 6: distancia máxima de edición de 2.
- $6 <$ largo ≤ 10: distancia máxima de edición de 3.
- $10 <$ largo: distancia máxima de edición de 4.

Hemos explicado cómo se lleva a cabo la búsqueda de candidatos a reemplazar una palabra errónea. Ahora resta ver de qué manera se elige la mejor opción entre los candidatos.

Como se mencionó anteriormente el método utilizado para la corrección de palabras implica la probabilidad de n-gramas.

Este método consiste en buscar la mayor probabilidad de ocurrencia de una palabra candidata en el contexto de la palabra con error. De acuerdo al resultado obtenido en la sección 5.3.1 para el parámetro *context_direction*, la mejor opción es utilizar todos los contextos (previous, middle y forward).

Con los contextos y las palabras candidatas se forman los n-gramas a los cuales los modelos de lenguaje le asignarán una probabilidad de ocurrencia.

Tomando la frase con error de un ejemplo anterior: "Se detuvo a senor en actitud sospechosa" y un conjunto de palabras candidatas a corregir el error $c_1,\ldots,c_n$.
Los n-gramas a obtener probabilidad de ocurrencia se formarán de la siguiente manera con i entre 1 y n:

Previous: "detuvo a c_i".

Middle: "a c_i en".

Forward: "c_i en actitud".

Para el caso del parámetro *context_direction* configurado como *all*, a cada palabra candidata se le calcula un *score*, dicho *score* se obtendrá de la suma de la probabilidad de cada contexto y será utilizado para elegir la palabra candidata más adecuada.

$$score(c_i) = P_{previous}(c_i) + P_{forward}(c_i) + P_{middle}(c_i)$$

Si el parámetro *context_direction* tiene una sola dirección en vez de *score* se utilizará la probabilidad del n-grama para el contexto dado.

Los modelos de lenguaje utilizados serán explicados en una sección siguiente.

El mecanismo de corrección de un documento consiste en la tokenización del texto para luego recorrerlo de izquierda a derecha, de arriba a abajo en busca de errores. Encontrado un error, se obtienen las palabras candidatas a corregirlo a través de *Symspell* y se forman los n-gramas según la configuración de contexto seleccionada. El modelo de lenguaje asigna probabilidades a estos n-gramas, la palabra candidata que forme n-gramas con mayor probabilidad o *score* calculado será la palabra candidata más adecuada a corregir el error. Estas correcciones se van agregando a medida que se recorre el texto, de esta manera las palabras siguientes podrán contar en su contexto con la palabra corregida anteriormente.

Para corregir el documento se realizan dos recorridas, en la primera se corrigen las palabras divididas por un guión y un salto de línea, mientras en la segunda se corrige el resto de palabras. Las palabras divididas pueden juntarse o ser marcadas para que en la siguiente recorrida no sean procesadas.

Podría ocurrir la situación en que un n-grama determinado tenga probabilidad nula según el modelo de lenguaje. Esto puede deberse a que el contexto contiene errores que no pudieron ser corregidos o contiene palabras muy poco comunes. En caso de que todos los n-gramas asociados a un error tengan probabilidad nula, se repite el procedimiento de selección de mejor candidato pero esta vez con un contexto más pequeño, si se utilizaban dos palabras en el contexto, ahora se utilizará una. Se calculan nuevamente las probabilidades, si existe algún n-grama con probabilidad no nula, se elige el de mayor probabilidad, de lo contrario se elimina el contexto y se busca la probabilidad únicamente de las palabras candidatas.

Modelos de n-gramas

Los modelos de lenguaje N-gramas calculan estadísticas de un corpus de texto para estimar la probabilidad de un nuevo enunciado en el lenguaje. Por esta razón, los modelos de n-gramas a menudo se usan en aplicaciones de PLN como el reconocimiento de voz, la traducción automática o como en nuestro caso la corrección de errores.

Google n-grams

Google n-grams es un inmenso corpus de n-gramas, si bien no es un modelo de lenguaje, la manera de utilizarlo será como si lo fuera. Como su nombre lo indica, fue creado por Google y es uno de los mayores recursos libres de lenguaje natural escrito. El conjunto de datos contiene miles de millones de frases con datos estadísticos extraídos de más de 8 millones de libros digitalizados por Google a través de OCRs. Para el idioma español se digitalizaron 854.649 libros con los cuales se contaron frecuencias de uno a cinco n-gramas.

n-gramas	Cantidad para español
1-grama	1.748.822
2-gramas	40.053.844
3-gramas	113.813.966
4-gramas	133.524.157
5-gramas	87.648.115

Tabla 5.10: Tabla de n-gramas.

Para poder consultar el enorme conjunto de datos (los datos comprimidos pesan nueve terabytes en total) se utiliza el motor de búsqueda PhraseFinder [33]. PhraseFinder, al estar indexado permite buscar rápidamente palabras o frases en el gigantesco conjunto de datos con un lenguaje de consulta fácil de usar. Esta herramienta está disponible online proporcionando una interfaz gráfica así como también una API.

PhraseFinder toma como entrada un n-grama y devuelve como salida la cantidad de veces que el n-grama ocurrió en el conjunto de datos, la cantidad de documentos que lo contienen y el año del documento más antiguo y el más actual que lo contiene. A su vez es *key sensitive* e indica el porcentaje de ocurrencia del n-grama en sus distintas variantes de mayúsculas.

Si se toma como ejemplo la frase "el policía que" se obtiene el resultado de la tabla Tabla 5.11 .

Frase	% variante	Ocur. indiv.	Ocur. en libros	Prim. y Últ. libro
el policía que	75,4 %	5.824	4.918	1871–2009
El policía que	23,5 %	1.814	1.553	1879–2009
el Policía que	1,2 %	90	74	1902-2009

Tabla 5.11: Tabla de n-gramas.

La Tabla 5.11 indica que de las distintas variaciones la opción más común para la frase utilizada es "el policía que" que ocurre 5.824 veces en el conjunto de datos. Esta herramienta es de suma importancia porque utilizando la cantidad de ocurrencias de n-gramas se puede obtener el n-grama más probable entre varios.

PhraseFinder fue integrado a nuestro proyecto a través de la API que proporciona, cuando se quiere corregir un error se invoca la API con las múltiples palabras candidatas incluido su contexto. Esta operación se paraleliza para disminuir los tiempos de obtención de probabilidades.

1Billion

Este modelo de lenguaje que denominamos 1Billion es un modelo de lenguaje construido a partir del corpus llamado *Spanish Billion Word Corpus and Embeddings* generado por un estudiante de doctorado en PLN [34]. Fue formado con la librería de Python KenLm [35] usando como entrada el popular corpus en español.

El anterior corpus del idioma español mencionado es no anotado y contiene casi 1.5 billones de palabras. 1Billion fue compilado de los siguientes corpus y recursos de la web libres:

- Parte en español de SenSem [36].
- Parte en español de Ancora Corpus[37].
- Tibidabo Treebank y IULA Spanish LSP Treebank [38].
- Parte en español de OPUS Project Corpora [39].
- Parte en español de Europarl (European Parliament) [40].
- Dumps de Wikipedia[41] en español, Wikisource [42] y Wikibooks [43], procesada con Wikipedia Extractor [44].

El estudiante quitó las anotaciones a los corpus Ancora, SenSem and Tibidabo y de los corpus paralelos de OPUS Project únicamente tomó las partes en español. Cambió los los caracteres no alfanuméricos por espacios, todos los números por el token "DIGITO" y por ultimo removió los múltiples espacios dejando únicamente un espacio.

Luego del procesamiento anterior el resultado fue un corpus con un total de:

- 1.420.665.810 palabras.
- 46.925.295 oraciones.
- 3.817.833 tokens únicos.

Obtenido el corpus, se utilizó la librería KenLm para formar el modelo de lenguaje.

ELMo

Embeddings from Language Models o ELMo es una representación contextualizada profunda de palabras en la cual se toma en cuenta las características complejas del uso de las palabras (semántica y sintaxis) y cómo varían según el contexto [45]. Puede ser usada fácilmente por cualquier modelo de lenguaje y se diferencia del resto de *word embeddings* en que a un token se le asigna una representación que es una función de toda la frase de entrada.

ELMo genera representaciones por medio de una red neuronal convolucional de dos capas y principalmente sirve para obtener *word embeddings*. Pero si se utilizan los pesos de la capa softmax al final de esta red se puede usar también como un modelo de lenguaje para predecir la probabilidad de la próxima palabra dado un contexto (o la anterior debido a que es un modelo de lenguaje bidireccional).

En el proyecto de grado de Rodrigo Lastra llamado "Inducción del Sentido de las Palabras para el Idioma Español" del año 2019, se entrenó ELMo para español con parte del vocabulario SBWC de la sección anterior, el cual se nos compartió para poder utilizar.

Para utilizar ELMo de la librería *allennlp.commands.elmo* se importa la clase *ElmoEmbedder*. Esta clase permite cargar los pesos obtenidos en el entrenamiento de ELMo para español realizado en el proyecto que fue compartido.

En una etapa anterior a considerar el uso del modelo de lenguaje 1Billion para nuestro proyecto, se realizaron pruebas preliminares para comparar el funcionamiento de ELMo frente a Google n-grams. Utilizando el preprocesamiento detallado en la sección 5.2 se comparó, usando la medida ratio, la corrección de 50 documentos elegidos al azar. Los resultados obtenidos estuvieron muy por debajo de los obtenidos usando Google n-grams, por lo tanto no se lo tuvo en consideración finalmente como modelo de lenguaje a utilizar y, a su vez, tampoco en la prueba de ajuste de parámetros de la sección 5.3.1.

Capítulo 6

Solución basada en traducción automática estadística

Pasaremos a explorar una solución alternativa al problema planteado en el presente proyecto basándonos en el paradigma de traducción automática. Si bien la aplicación natural de los traductores automáticos es la de traducir textos de un idioma determinado a otro distinto, la corrección de errores en documentos generados por medio de OCRs puede abordarse como el problema de traducir texto incorrecto a texto correcto sobre un mismo idioma. Bajo estas condiciones el idioma o lenguaje de origen sería el conformado por las salidas del OCR mientras que el idioma destino estaría dado por las reconstrucciones de los documentos a partir de los datos recabados en la plataforma LUISA. Este enfoque ya ha sido utilizado en varios trabajos [16] [46].

6.1. Construcción de corpus paralelo

Para el entrenamiento de un modelo de traducción se necesita un corpus paralelo, dicho corpus está conformado por un conjunto de textos junto con sus traducciones, donde además los datos se encuentran alineados a cierto nivel de granularidad; como puede ser a nivel de renglón, oración o párrafo.

Como se dijo anteriormente, el lenguaje de origen serán las salidas del OCR y el de destino (también podemos llamarlo traducción) serán las reconstrucciones a partir de LUISA. Entonces contamos con documentos de texto y sus traducciones, el único inconveniente para poder considerarlo un corpus paralelo es el alineamiento.

¿Por qué es necesario alinear? Recordemos que contamos con dos conjuntos de textos, que si bien son generados a partir de los mismos documentos digitales, estos pueden variar en cierta medida a nivel de alineación. Como ya hemos visto, esto se debe principalmente a que los textos procesados por el OCR pueden contener diversos tipos de errores, entre los que se encuentran el agregado de saltos de líneas inexistentes; ya sea porque se interpretó una única línea del documento digitalizado como varias líneas distintas o porque se detectó algún tipo de ruido (por ejemplo manchas) en espacios vacíos. A su vez, la estructura de los textos a partir de las reconstrucciones utilizando los datos de LUISA tampoco es cien por ciento fiel a la estructura de los documentos originales.

Teniendo en cuenta la situación descrita, se necesita alinear las parejas de documentos para así construir un corpus paralelo con el que entrenar el modelo de traducción. A la hora de definir el nivel de granularidad con el que alinear (unidad de alineación o UA) optamos por dos opciones distintas:

1. Alineación por oraciones
2. Alineación por renglones [1]

Alineación por oraciones

La motivación para haber tomado este camino se basó en que la plataforma que se utilizó para construir el sistema de traducción recomienda usar esta UA a la hora de definir los ejemplos de entrenamiento. Idealmente, requiere que cada ejemplo de entrenamiento esté conformado por una pareja de oraciones, siendo una de ellas una oración perteneciente al texto origen y la oración restante su traducción asociada en el texto destino.

Para construir un conjunto de ejemplos con las características mencionadas, en primer lugar es necesario identificar y aislar cada oración de ambos textos (segmentación en oraciones) para pasar luego al proceso de alineación.

Analicemos un ejemplo de estos pasos para que quede más claro. En las tablas 6.1 y 6.2 podemos ver dos ejemplos ficticios de texto generado por OCR y texto transcrito de forma manual respectivamente. Estos ejemplos fueron construidos buscando representar en breves fragmentos las distintas situaciones con las que nos encontramos en los textos originales. Podemos notar que una línea no tiene por qué corresponderse 1 a 1 con las oraciones de un texto, algunas pueden contener más de una oración, o una oración puede extenderse por más de una línea. Además, se aprecia que una línea completa en el texto del OCR (línea 2 de la Tabla 6.1) está ocupada por texto que no tiene sentido, lo que quiere decir que existen líneas en uno u otro texto que pueden no tener una traducción asociada.

Línea	Texto Origen
1	*Maf0 s luz -un asfuego abandono. SUbe aAsu canto una págar0 enamorado.*
2	*.....00asd00———-.*
3	*Yantaz,, criaturas- ávides eeqwn mi silencio y e pequeña llluvia ue me acom-*
4	*pana sin caer. M3 4detenjo A rdfespiro.*

Tabla 6.1: Ejemplo de texto origen antes de segmentar por oraciones

Línea	Texto Destino
1	*Mata su luz un fuego abandonado. Sube su canto un pájaro enamorado.*
2	*Tantas criaturas ávidas en mi silencio y esta pequeña lluvia que me acom-*
3	*paña sin cesar. Me detengo y respiro.*

Tabla 6.2: Ejemplo de texto destino antes de segmentar por oraciones

Entonces, en primera instancia realizamos una segmentación en oraciones tanto en el texto origen como en el texto destino, siempre manteniendo el orden de estas. Esta segmentación, y en general todas las segmentaciones llevadas a cabo en el proyecto, se realizaron utilizando la plataforma *Natural Language Toolkit* [47], más conocida como NLTK, la cual facilita entre otros un paquete llamado

[1] Entiéndase por renglón a un fragmento de texto ubicado en una misma horizontal dentro de un documento. Si nos centramos en el contexto informático, aquí los renglones serían el conjunto de caracteres que se encuentran delimitados entre dos caracteres especiales de **salto de línea**.

tokenize dónde se incluye la función *sent_tokenize()* cuyo propósito es el de segmentar los textos en oraciones. Se decidió trabajar con esta plataforma ya que cuenta con una enorme cantidad de herramientas y libertades para el manejo de textos y tareas de PLN, además de ser una de las más reconocidas contando con un nivel de soporte y actualización muy alto.

En las tablas 6.4 y 6.5 podemos ver los textos resultantes luego de la segmentación en oraciones. En ambos casos se obtuvo un número de líneas mayor al original. Notemos que las oraciones ocupando un mismo número de línea no tienen por qué corresponderse entre ambos textos: por ejemplo, la oración de la línea 4 del texto destino (Tabla 6.5) no es la traducción de la oración que se encuentra en la línea 4 del texto origen (Tabla 6.4). Será trabajo del algoritmo de alineación el encontrar las correspondencias entre oraciones, para así formar las tuplas o ejemplos de entrenamiento.

Línea	Texto Origen
1	*Maf0 s luz -un asfuego abandono.*
2	*SUbe aAsu canto una págar0 enamorado.*
3	*.....00asd00———-.*
4	*Yantaz„ criaturas- ávides eeqwn mi silencio y e pequeña llluvia ue me acom- pana sin caer.*
5	*M3 4detenjo A rdfespiro.*

Tabla 6.4: Ejemplo de texto origen luego de segmentar por oraciones

Línea	Texto Destino
1	*Mata su luz un fuego abandonado.*
2	*Sube su canto un pájaro enamorado.*
3	*Tantas criaturas ávidas en mi silencio y esta pequeña lluvia que me acom- paña sin cesar.*
4	*Me detengo y respiro.*

Tabla 6.5: Ejemplo de texto destino luego de segmentar por oraciones

Alineación por renglones

Notamos que el proceso de segmentación en oraciones utilizado en el enfoque anterior era propenso a cometer errores dada la naturaleza de los datos. Sobre todo en el texto generado por el OCR existe un nivel de ruido bastante alto, en el que se introducen muchos signos de puntuación, en su mayoría puntos de fin de oración, que evidentemente dificultan una segmentación en oraciones, ya que actúan como delimitadores de oración. A su vez, la misma estructura de los documentos (en muchos casos listas que carecen de signos de puntuación), inducen al error en el proceso de segmentación resultando en oraciones largas donde en realidad no lo había. Por esta razón, la cantidad de segmentos del texto origen y la cantidad de segmentos de su traducción diferían en gran medida, dificultando la alineación. Entonces, para este enfoque alternativo de alineación por renglones, se decidió tomar como UA a las líneas, teniendo en cuenta que generalmente la cantidad de líneas en ambos textos es similar y en el mismo orden. Para este enfoque no hizo falta ningún tipo de preprocesamiento.

6.1.1. Proceso de Alineación

Para realizar la alineación, se evaluó la posibilidad de implementar un algoritmo haciendo uso de la medida *ratio* de *SequenceMatcher*, la misma que es utilizada para comparar los resultados de los OCRs analizados y en general todas las comparaciones de resultados en este proyecto. Con esta opción, se podrían tomar en cuenta todos los detalles de los documentos y transcripciones de nuestro

universo y posiblemente obtener un mejor resultado, pero se entendió que el problema presentaría muchos detalles (performance, ordenaciones, evaluación de correcta funcionalidad, etc.) que harían compleja la solución, por lo que se resolvió utilizar alguna herramienta ya disponible.

Tras investigaciones y pruebas, entendimos adecuada la utilización de la herramienta llamada Bleualign [48], que se basa en la similitud (puntuación BLEU) entre las UAs del texto origen y las UAs de texto de destino para realizar la alineación. Los principales motivos por los que se decidió utilizar esta herramienta respecto a otras son: que es una herramienta de código abierto, lo que nos permitió entender el proceso de alineación y así evaluar si era adecuada utilizarla, y que es posible configurar fácilmente distintas variables permitiendo adaptarse al problema particular.

Bleualign

Bleualign es una herramienta de código abierto desarrollada en la *Universidad de Zurich*. La herramienta permite alinear UAs de dos textos paralelos en idiomas diferentes, para esto realiza el cálculo del puntaje BLEU entre el texto origen (traducido al idioma objetivo mediante cualquier herramienta de traducción) y el texto objetivo.

En nuestro caso, no aplica tener ni hacer uso del texto origen traducido, ya que ambos textos (origen y destino) están en el mismo idioma, por lo que se utilizó el texto origen sin traducir para el cálculo de puntajes BLEU. Por otra parte, vale mencionar que la herramienta está enfocada para generar alineaciones a nivel de oraciones en los párrafos, sin embargo para uno de los dos enfoques que se usó, se aplicó para resolver alineación entre líneas.

BLEU como puntaje de similitud

Como ya se adelantó en el capítulo 2.4, BLEU es un método desarrollado para medir de forma automática la calidad de la traducción de los sistemas MT, comparando la traducción del sistema con una o más traducciones de referencia [11]. Esto se realiza midiendo la precisión de n-gramas de la traducción del sistema (hipótesis) para todos los niveles de n-gramas hasta cuatro. En resumen, cuenta la cantidad de n-gramas, desde unigramas hasta cuatrigramas, que coinciden entre la traducción generada por el sistema MT y la o las traducciones de referencia. En la publicación referida a la herramienta afirman y evidencian que obtuvieron mejores resultados calculando la precisión de n-gramas para todos los niveles hasta 2 (unigramas y bigramas), en vez de hasta cuatrigramas como es calculada la puntuación BLEU generalmente. Esto es porque cuánto más grande sea el límite de n-gramas, más probabilidad hay de que el puntaje sea 0, en parte debido al bajo rendimiento del sistema MT que utilizaron para generar el texto origen. Para nuestro problema, luego de realizar pruebas, también encontramos mejor calcular la precisión de n-gramas contando hasta bigramas. Esto es razonable teniendo en cuenta que el texto origen de nuestro problema, tiene muchos errores por ser una salida obtenida de un sistema de OCR sobre imágenes de baja calidad.

Algoritmo de alineación implementado en Bleualign

Siendo S y T el conjunto de UAs del texto origen y objetivo respectivamente, el resultado de la alineación se puede ver como el conjunto de parejas (s,t), con $s \in S$ y $t \in T$, donde cada pareja representa la alineación de una UA del texto origen con una del texto objetivo. En el caso trivial, la alineación entre dos textos es (s_1, t_1), (s_2, t_2), ..., (s_n, t_n).

El algoritmo de alineación consta de dos pasos:

1. Se identifica un conjunto de puntos de anclaje (los cuales representan las alineaciones más evidentes) usando la puntuación BLEU como medida de similitud entre el texto origen y el texto objetivo. Para obtener los puntos de anclaje, se calcula el puntaje BLEU de todos los pares posibles (producto cartesiano de S y T) que tengan al menos un bigrama en común [2]. Para el conjunto de pares obtenidos, se realiza una poda manteniendo solamente los 3 candidatos de alineación con mejor puntaje para cada UA del texto origen. Luego se busca el conjunto de alineaciones 1 a 1 que maximiza la puntuación en el documento, sin violar el orden monotónico de los pares de UAs, o sea sin definir alineaciones cruzadas, o lo que es lo mismo, que los segmentos A y B en el texto origen no corresponderán a los segmentos B'A' en el texto objetivo. El resultado de este paso es una lista ordenada de alineaciones 1 a 1, mientras que varias UAs en S y T permanecen sin alinear.
2. Para todos los pares de alineaciones 1 a 1 (s_i, t_j), (s_k, s_l) resultado del paso anterior, se extraen todas las UAs no alineadas s_x y t_y para las cuales $i < x < k$ y $j < y < l$. Del mismo modo, se extraen todas las UAs no alineadas antes del primer y después del último índice de los puntos de anclaje. Estas UAs (llamadas huecos) se alinean usando las heurísticas bleu1to1 y luego el algoritmo basado en la longitud de Gale & Church explicadas más adelante. El algoritmo de Gale & Church [49] busca posibles alineaciones basándose en el principio de que las UAs equivalentes deben corresponder aproximadamente en longitud.

La herramienta también permite modificar algunas variables útiles para distintos casos. Algunas de las variables más relevantes son:

- bleu_ngrams {entero}: Se utiliza para definir la cantidad de tokens utilizados para calcular el puntaje bleu (ej: unigramas, bigramas, etc.)
- Nto1 {entero}: Indica a cuantas UAs en el texto origen puede estar relacionada una UA en el texto objetivo, y al revés.
- gapfillheuristics {[bleu1to1, galechurch]}: Define qué método o métodos utilizar para alinear los huecos.
    - bleu1to1: Se agrega el par (s_i, t_j) al conjunto solución, si maximiza la puntuación bleu. Especialmente en textos más grandes, es posible que el par (s_i, t_j) se haya eliminado en la poda y, por lo tanto, no se haya considerado en el algoritmo de búsqueda del conjunto que maximiza el puntaje.
    - galechurch: Si el hueco es de tamaño 0 en cualquiera de los dos textos, o si su tamaño es asimétrico por un factor mayor que dos, las UAs restantes en el hueco quedan sin alinear. De lo contrario, el proceso de alineación basado en la longitud de Gale & Church establece una alineación entre todas las UAs restantes en el hueco.

Se puede ver un pseudocódigo del algoritmo en la sección 10.3 del anexo.

[2] Al menos en común el mayor nivel de n-grama a comparar configurado, bigrama en nuestro caso

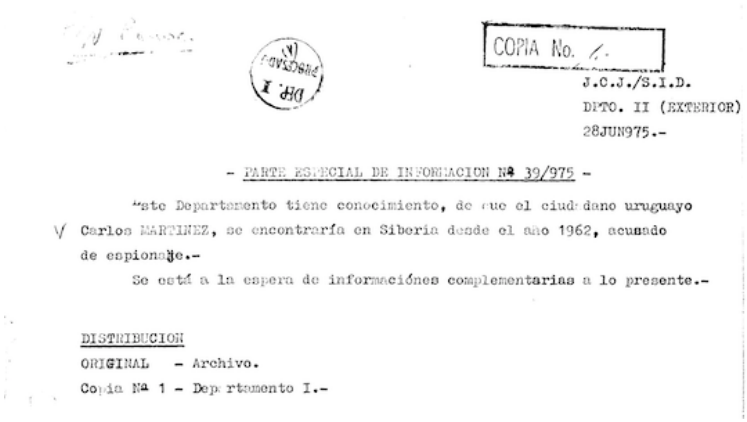
Ejemplo

Luego de varias pruebas con distintas configuraciones, la que dio mejores resultados para nuestro problema fue:

- bleu_ngrams: 2
- Nto1: 1
- gapfillheuristics: bleu1to1

Con esta configuración (en particular no incluir galechurch en gapfillheuristics), notamos que algunas alineaciones incorrectas se descartaban, como así también algunas correctas, pero la razón principal por la que se optó usar esta es que se entendió que para el entrenamiento del sistema de traducción automático sería mejor tener menos datos en total, con la menor cantidad de alineaciones incorrectas, a tener mayor cantidad total de datos con más alineaciones incorrectas. Esta configuración es la utilizada a lo largo del proyecto y en el ejemplo que se muestra a continuación.

Para este ejemplo utilizamos el documento que se muestra en la Imagen 10.2 del anexo. Adjuntamos también una parte recortada (Imagen 6.1) recortando la parte más interesante del documento para poder ver más fácilmente los resultados. Se busca alinear los textos a nivel de líneas.



COPIA No. 1.

J.C.J./S.I.D.
DPTO. II (EXTERIOR)
28JUN975.-

- PARTE ESPECIAL DE INFORMACION Nº 39/975 -

"ste Departamento tiene conocimiento, de que el ciudadano uruguayo Carlos MARTINEZ, se encontraría en Siberia desde el ano 1962, acusado de espionaje.-

Se está a la espera de informaciónes complementarias a lo presente.-

DISTRIBUCION
ORIGINAL - Archivo.
Copia Nº 1 - Departamento I.-

Imagen 6.1: Documento de ejemplo

La salida resultado del OCR para el documento (Texto origen), y la transcripción obtenida mediante LUISA (Texto objetivo) se muestran en la Tabla 6.6. Se puede apreciar que el texto de LUISA tiene más líneas que el del OCR, por lo tanto, debido a cómo está implementada la solución, al menos va a ser necesario descartar algunas líneas. También es interesante ver cómo el OCR descartó los bordes de la hoja y sellos que se pueden ver en la Imagen 10.2, pero no en la Imagen 6.1.

	Texto origen	Texto objetivo	
s_1	J .C. J./S.I.D.	COPIA	t_1
s_2	DITO. II (EXTERIOR)	I . J.C.J./S.I.D.	t_2
s_3	28JUN975.-	I DEP	t_3
s_4	- ESPECIAL DE INFORILACION H4 39/975 -	DPTO. II ()EXTERIOR	t_4
s_5	^ste Departamento tiene conocimiento, de ruc el ciud* daño uruguayo	JUN975.-	t_5
s_6	\/ Carlos IL'iRTINEZ, se encontraría en Siberia desde el alio 1962, acusado	PARTE -	t_6
s_7	de espionaje.-	Este Departamento tiene conocimiento, de que el ciudadano uruguayo	t_7
s_8	Se está a la espera de informaciónes complementarias a lo presente.-	V Carlos MARTINEZ, se encontraría en Siberia desde el año 1962, acusado	t_8
s_9	DISTRIBUCION	de espionaje.-	t_9
s_{10}	ORIGINAL - Archivo.	Se está a la espera de informaciónes complementarias a lo presente.-	t_{10}
s_{11}	Copia NA 1 - Dep, rtamento I.-	DISTRIBUCION	t_{11}
		ORIGINAL - Archivo.	t_{12}
		Copia No 1 - Departamento I.-	t_{13}
		M	t_{14}
		A/N	t_{15}
		No. de Orden 1215	t_{16}

Tabla 6.6: Textos obtenidos mediante el OCR y las transcripciones en LUISA para el documento de la Imagen 10.2 indexadas respectivamente

Paso 1: En el primer paso se obtienen los pares de líneas con bigramas en común, el cálculo de los puntajes se muestra en Tabla 6.7, cabe destacar que no se muestran los puntajes de alineaciones que se puedan haber eliminado en la poda. Luego se establecen los puntos de anclaje como el conjunto de alineaciones que maximiza el puntaje Bleu (Tabla 6.8). Se puede observar en la Tabla 6.7 que existen conflictos en el orden monotónico, en la mayoría de los casos con la línea t_{13} en líneas del texto origen más cercanas al índice 1. Este conflicto se resuelve buscando el conjunto de pares que sumen el puntaje más alto, en este caso el de la Tabla 6.8, descartando (s_1, t_{13}), (s_3, t_9), (s_3, t_{13}), (s_7, t_5), (s_7, t_{13}), (s_8, t_{13}), (s_8, t_9), (s_{11}, t_9) y (s_{11}, t_{10})

Pares	Puntaje
(s_1, t_2)	0.860
(s_1, t_{13})	0.107
(s_2, t_4)	0.577
(s_3, t_5)	0.577
(s_3, t_9)	0.411
(s_3, t_{13})	0.138
(s_5, t_7)	0.418
(s_6, t_8)	0.647
(s_7, t_9)	1.0
(s_7, t_5)	0.411
(s_7, t_{13})	0.167
(s_8, t_{10})	1.0
(s_8, t_{13})	0.107
(s_8, t_9)	0.76
(s_{10}, t_{12})	1.0
(s_{11}, t_{13})	0.444
(s_{11}, t_9)	0.113
(s_{11}, t_{10})	0.112

Tabla 6.7: Pares de alineaciones entre dos textos dados por el puntaje BLEU. Líneas identificadas por su índice.

Pares
(s_1, t_2)
(s_2, t_4)
(s_3, t_5)
(s_5, t_7)
(s_6, t_8)
(s_7, t_9)
(s_8, t_{10})
(s_{10}, t_{12})
(s_{11}, t_{13})

Tabla 6.8: Conjunto de alineaciones que maximiza el puntaje Bleu

Paso 2: Los huecos resultantes de aplicar el paso 1 y quedarse con los puntos de anclaje se muestran en la Tabla 6.9. En este caso, el cálculo repetido de la puntuación Bleu (configuración bleu1to1 de la variable gapfillheuristics) para los huecos no muestra nuevas alineaciones, lo que significa que la poda no había descartado ninguna posible alineación. No se evalúan nuevas alineaciones utilizando el algoritmo de Gale & Church, dado que la configuración utilizada indica que no se realice ese procesamiento.

	Texto origen	Texto objetivo	
s_4	- ESPECIAL DE INFORILACION H4 39/975 -	COPIA	t_1
s_9	DISTRIBUCION	I DEP	t_3
		PARTE -	t_6
		DISTRIBUCION	t_{11}
		M	t_{14}
		A/N	t_{15}
		No. de Orden 1215	t_{16}

Tabla 6.9: Huecos resultantes del paso 1 del ejemplo 1

Resultados

En la Tabla 6.10 se muestran las líneas resultado de la alineación. Se puede afirmar que las alineaciones obtenidas son todas correctas, dado que los elementos de los pares son muy parecidos y en algunos casos iguales. El caso que llama la atención es el del par (s_9, t_{11}) que es claro que son la misma línea. La razón por la que este caso queda sin cubrir es porque el puntaje Bleu se calcula únicamente en líneas que tengan al menos un bigrama en común, esto tiene sentido en la herramienta ya que está pensado para alinear oraciones en lugar de líneas. Este caso sí se cubre si también se asignara galechurch en la variable gapfillheuristics, pero también se introduciría una alineación incorrecta respecto a la línea s_4 como se ve en la Tabla 6.11 (línea p_4') que muestra el resultado de la alineación si también se hubiera usado el algoritmo Gale & Church en el paso 2.
Aprovechando que la herramienta es de código abierto, el algoritmo podría ser modificado para contemplar los detalles del problema en cuestión, pero se entendió que entrar en ese problema se alejaba del objetivo principal del proyecto y lo alargaría demasiado, por lo tanto se decidió usarla sin modificar y proponer la mejora como trabajo a futuro.

	línea origen	Línea objetivo
p_1	J .C. J./S.I.D.	I . J.C.J./S.I.D.
p_2	DITO. II (EXTERIOR)	DPTO. II ()EXTERIOR
p_3	28JUN975.-	JUN975.-
p_4	^ste Departamento tiene conocimiento, de ruc el ciud* daño uruguayo	Este Departamento tiene conocimiento, de que el ciudadano uruguayo
p_5	\/ Carlos IL'iRTINEZ, se encontraría en Siberia desde el alio 1962, acusado	V Carlos MARTINEZ, se encontraría en Siberia desde el año 1962, acusado
p_6	de espionaje.-	de espionaje.-
p_7	Se está a la espera de informaciónes complementarias a lo presente.-	Se está a la espera de informaciónes complementarias a lo presente.-
p_8	ORIGINAL - Archivo.	ORIGINAL - Archivo.
p_9	Copia NA 1 - Dep, rtamento I.-	Copia No 1 - Departamento I.-

Tabla 6.10: Resultado de la alineación de los textos de la Tabla 6.6 utilizando la configuración presentada

	línea origen	Línea objetivo
p_1'	J .C. J./S.I.D.	I . J.C.J./S.I.D.
p_2'	DITO. II (EXTERIOR)	DPTO. II ()EXTERIOR
p_3'	28JUN975.-	JUN975.-
p_4'	- ESPECIAL DE INFORILACION H4 39/975 -	PARTE -
p_5'	^ste Departamento tiene conocimiento, de ruc el ciud* daño uruguayo	Este Departamento tiene conocimiento, de que el ciudadano uruguayo
p_6'	\/ Carlos IL'iRTINEZ, se encontraría en Siberia desde el alio 1962, acusado de espionaje.-	V Carlos MARTINEZ, se encontraría en Siberia desde el año 1962, acusado de espionaje.-
p_7'	Se está a la espera de informaciónes complementarias a lo presente.-	Se está a la espera de informaciónes complementarias a lo presente.-
p_8'	DISTRIBUCION	DISTRIBUCION
p_9'	ORIGINAL - Archivo.	ORIGINAL - Archivo.
p_{10}'	Copia NA 1 - Dep, rtamento I.-	Copia No 1 - Departamento I.-
p_{11}'		

Tabla 6.11: Resultado de la alineación de los textos de la Tabla 6.6 agregando galechurch como valor a la variable gapfillheuristics de la configuración

6.1.2. Resultados de alinear los datos

En esta sección mostraremos los resultados de la alineación para los dos enfoques (alineación por oraciones y alineación por renglones) a través del conteo de líneas que permanecen en los textos. Contar el número de líneas remanentes es útil dentro de ambos enfoques: En la alineación por renglones es evidente, ya que cada línea puede conformar potencialmente un ejemplo de entrenamiento. Por otro lado, en la alineación por oraciones, ocurre que luego de segmentar en oraciones se colocó una oración por línea, por lo que caemos en la misma situación que el enfoque alternativo de renglones.

A modo ilustrativo, si el texto de origen y destino terminaron con 100 líneas cada uno, entonces tendríamos potencialmente 100 ejemplos para entrenar. Decimos potencialmente pues más adelante se verá que se aplicaron ciertos preprocesamientos a los textos los cuales pueden reducir el número de ejemplos. En el diagrama de la Imagen 6.2 se especifican los pasos seguidos los cuales serán explicados con más detalle en el correr de esta sección.

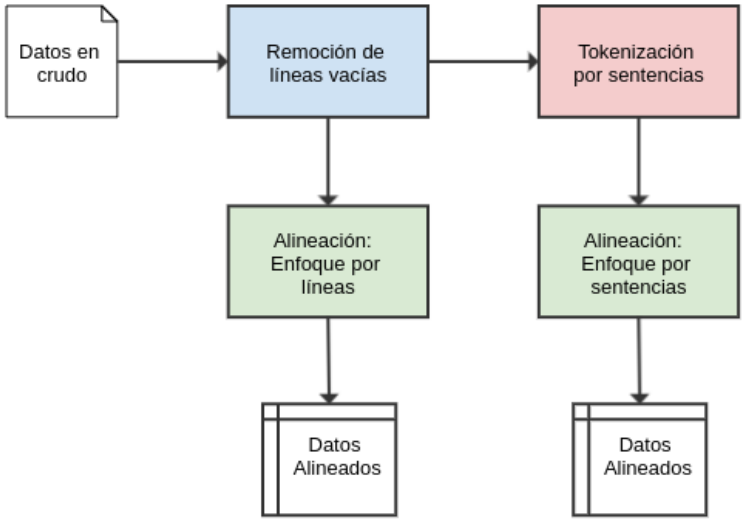


Imagen 6.2: Proceso de alineación para ambos enfoques

Yendo a los números previo a la alineación, una vez unidos todos los documentos generados por el OCR entre sí por un lado, y todos los documentos reconstruidos gracias a los datos de LUISA por otro, obtenemos el siguiente número de líneas visible en la Tabla 6.12.

Vemos una diferencia notoria de más del doble de líneas a favor del texto del OCR, o texto origen. Esto se debe principalmente a dos factores. El primero es que con frecuencia el OCR toma en cuenta

	LUISA	OCR
Nº de líneas	36396	74420

Tabla 6.12: Número total de líneas en ambos textos

líneas vacías y las procesa como líneas en blanco, mientras que en los textos de LUISA no es común que se produzcan renglones vacíos. En segundo lugar, muchas veces el OCR interpreta pequeñas manchas o imperfecciones en las imágenes como caracteres, agregando líneas con datos dónde en realidad no había nada.

A continuación se pasó a eliminar las líneas vacías. El resultado se puede apreciar en la Tabla 6.13. Es muy importante recalcar que estos dos textos (una vez eliminadas las líneas vacías) fueron los que se tomaron como entrada para el proceso de alineación en el enfoque de líneas o renglones, no así para el enfoque de alineación por oraciones, ya que aún falta segmentar en oraciones, como se verá más adelante.

	LUISA	OCR
Nº de líneas	34342	42545

Tabla 6.13: Número de líneas luego de remover líneas vacías

Comparando con los datos de la Tabla 6.12, se ve claramente que se removieron en el orden de las 30 mil líneas vacías en el texto del OCR, y poco más de 2 mil líneas en el caso de LUISA, corroborando que el número de líneas vacías que agrega el OCR supera en gran cantidad a las de LUISA.

Ahora veamos en que condición quedaron los textos habiendo tokenizado el resultado anterior por oraciones, manteniendo una oración por línea (Tabla 6.14). En esta oportunidad, los textos fruto de este paso pasarán a ser la entrada del proceso de alineación por oraciones.

	LUISA	OCR
Nº de líneas	23514	20140

Tabla 6.14: Número de líneas luego de tokenizar por oraciones

La cantidad de líneas se redujo considerablemente tanto en el texto origen como el texto destino, evidenciando que en un gran número de casos una oración ocupa más de una línea, por ejemplo, si una oración ocupaba tres líneas, ahora pasó a ocupar una única línea de mayor largo. Este hecho está ligado obviamente al ancho de los documentos originales en los que se escribieron los textos, el cual supone una cota superior a la cantidad de caracteres que puede contener una línea. También, como ya se ha mencionado anteriormente, existen muchos textos que contienen listas, por lo que cuentan con estructuras de pocas palabras entre varias líneas, sin ningún tipo de delimitador de oraciones entre los elementos de la lista (como podrían ser los puntos de fin de oración). Esto último lleva a que muchas líneas converjan en una sola luego de tokenizar.

Finalmente veamos que ocurre luego de alinear los textos. Recordemos que para el enfoque por oraciones utilizamos los textos descritos en la Tabla 6.14, mientras que para el enfoque por renglones empleamos los datos del paso previo (ver Tabla 6.13). En esta ocasión alcanza con contar el número de líneas de un único texto para cada enfoque, ya que posterior a la alineación de ambos textos, origen y destino cuentan con la misma cantidad de líneas. El resultado se puede ver en la Tabla 6.15

Se puede apreciar que en el enfoque de alineación a nivel de líneas se cuenta con una mayor cantidad de líneas como era de prever. Es entonces que potencialmente se tienen 6562 ejemplos para trabajar

	Enfoque por Oraciones	Enfoque por Líneas
Nº de líneas	6562	17460

Tabla 6.15: Número de líneas en cada enfoque luego de alinear

en el entrenamiento del sistema de traducción bajo el enfoque de alineación por oraciones y 17460 ejemplos considerando el enfoque restante. Nuevamente hablamos de una cantidad potencial de ejemplos, ya que en el proceso de entrenamiento del sistema de traducción automática existe la posibilidad de remover ciertos ejemplos tomando en cuenta distintas características, como pueden ser el largo de línea calculado tanto a nivel de palabras como de caracteres.

En la siguiente sección pasaremos a describir el proceso de construcción del traductor automático tomando como inputs los textos alineados.

6.2. Construcción del traductor

El sistema de traducción fue construido utilizando Moses [50], software gratuito que permite entrenar modelos para la traducción automática estadística. En esta sección describiremos el *framework* así como el entrenamiento realizado.

6.2.1. Moses

Moses es un software *opensource* que brinda un *toolkit* completo para la construcción de un SMT. Entre las herramientas que lo conforman encontramos: módulos para limpieza y preparación de corpus, software de alineación a nivel de palabras, implementaciones de medidas para evaluación de resultados y módulos para la construcción de modelos del lenguaje. No todas estas herramientas son implementadas por el *framework*, sino que algunas son herramientas externas como GIZZA++ [51] para alineación de palabras.

Esta clase de sistemas son entrenados utilizando corpus paralelos con grandes cantidades de datos a partir de los cuales el sistema aprende a traducir pequeños segmentos de texto. El proceso de entrenamiento de Moses toma los datos paralelos y utiliza las coincidencias de palabras y segmentos para inferir las correspondencias de traducción entre los dos idiomas de interés. En traductores basados en frases, como el que fue construido para esta tarea, las correspondencias se dan simplemente entre secuencias contiguas de palabras. El modelo presentado por Moses fue definido en el artículo *Statistical Phrase-Based Translation* [52].

Además del entrenamiento con el corpus paralelo, parte de la construcción de este sistema implica la creación de un modelo de lenguaje para el idioma objetivo (en este caso el idioma español) de dónde el sistema aprende cómo lucirá el idioma al que queremos llegar. En lugar de entrenar un nuevo modelo de lenguaje, decidimos reutilizar el modelo de lenguaje entrenado para la solución anterior con el corpus "Billion Word Corpus", por lo cual el proceso descrito de aquí en más trabajará procesando el corpus paralelo y no abordará la construcción de un modelo de lenguaje.

El sistema está conformado por dos componentes principales: el **pipeline de entrenamiento** y el **decodificador**. El pipeline de entrenamiento es una colección de herramientas que toman los datos en crudo (tanto del corpus paralelo como del corpus monolingüe) y construyen un modelo de traducción. Por otra parte, el decodificador es una única aplicación escrita en C++ la cual, dado un modelo de traducción ya entrenado y una frase en el idioma origen, se encarga de traducir la frase al idioma destino.

6.2.2. Conjunto de datos

Como fue explicado en la sección 6.1, se tomaron dos enfoques a la hora de alinear los textos origen y destino para construir los corpus paralelos. Esto nos dejó con 2 conjuntos de datos (cada uno con su correspondiente pareja de textos origen-destino) para trabajar en la construcción del SMT. A lo largo de la descripción del proceso de entrenamiento que se detalla a continuación iremos mostrando los cambios que sufren cada uno de los dos conjuntos iniciales de datos siempre que sea pertinente.

A partir de ahora nos referiremos como *conjunto-O* al conjunto de datos obtenidos luego de aplicar el *enfoque de alineación por oraciones*, y como *conjunto-R* a los datos resultantes de haber seguido el *enfoque de alineación por renglones*.

También creemos importante dar una noción más formal de a qué nos referimos al hablar de un *ejemplo de entrenamiento* particular en el contexto de construcción del traductor. Desde ahora un *ejemplo de entrenamiento* está formado por la pareja de oraciones (s,t) donde s es una oración perteneciente al texto origen (generado por el OCR) y t es su traducción asociada dentro del texto destino (reconstrucciones de LUISA).

Bajo estas aclaraciones, podemos decir que el conjunto-O cuenta con 6562 ejemplos de entrenamiento mientras que el conjunto-R posee 17460 ejemplos. Un detalle no menor a destacar es que el número de ejemplos contenido en cualquiera de los dos grupos es tremendamente pequeño comparado con lo que comúnmente se utiliza en la práctica al buscar buenos resultados. La guía de Moses brinda 130.000 ejemplos para el traductor allí construido y aclara que es un conjunto chico. En los trabajos relacionados que hemos investigado, los números de ejemplos alcanzan el orden de millones.

Un último punto a definir que puede resultar ambiguo antes de comenzar con el repaso del entrenamiento y la decodificación es qué se entiende por el concepto de frase: se considera frase a una secuencia de palabras consecutivas, entonces una frase no necesariamente tiene que ser una oración completa.

6.2.3. Pipeline de entrenamiento

El entrenamiento en Moses está conformado por varios pasos implementados a modo de pipeline como el nombre de la sección sugiere.

Cabe resaltar que la mayoría de los pasos de aquí en adelante se ejecutaron utilizando las herramientas provistas por Moses. El único paso implementado por nosotros fue el paso número 2.

Los pasos del entrenamiento se listan a continuación.

1) Limpieza de datos

En primera instancia realizamos un preprocesamiento de los datos (tanto para el conjunto-O como el conjunto-R), ya que estos necesitan ser preparados previo a su utilización en el entrenamiento, para ello aplicamos los siguientes tres algoritmos: tokenización, truecasing y cleaning.

- **Tokenización**: Es una tokenización a nivel de palabra. Se insertan espacios entre cada palabra y signo de puntuación, por lo tanto cada token queda determinado por los espacios en blanco de sus extremos. Además todo grupo de espacios en blanco consecutivos es reducido a un único espacio.

- **Truecasing**: Consiste en determinar el correcto uso de mayúsculas en las palabras de los textos. El proceso ejecutado aquí normalmente mantiene todas las palabras como están a excepción de la

palabra inicial de cada oración, la cual podría estar en minúscula en cuyo caso lo más probable es que su primer letra se pase a mayúscula.

- **Cleaning**: En este paso se eliminan los siguientes tipos de líneas junto a sus traducciones asociadas:
  - Líneas vacías (recordemos que las líneas vacías ya habían sido eliminadas al generar los corpus paralelos, por lo que esta característica de la funcionalidad no tiene ningún efecto)
  - Líneas que superen un largo de 1000 caracteres
  - Líneas que superen las 80 palabras

Los procesos de tokenización y truecasing no afectan el número final de ejemplos, sólo realizan alteraciones sobre los mismos. Este no es el caso del proceso de cleaning, el cual es capaz de remover ejemplos. En la Tabla 6.16 podemos ver la cantidad ejemplos luego de haber ejecutado el proceso de limpieza sobre cada conjunto.

	Conjunto-O	Conjunto-R
Pre-limpieza	6562	17460
Post-limpieza	6070	17447

Tabla 6.16: Número de ejemplos antes y después del proceso de limpieza

2) División en subconjunto de entrenamiento y evaluación

Una vez preparados los datos y antes de comenzar efectivamente a entrenar un modelo es necesario separar una porción de los datos para utilizarlos únicamente en la evaluación. Siguiendo los lineamientos de las soluciones anteriores, decidimos dividirlos en una proporción 70/30, dónde 70 % constituirá el nuevo conjunto de entrenamiento y el restante 30 % será utilizado para evaluación.

La idea del algoritmo que se implementó para llevar a cabo la división constó de los siguientes pasos:

1. Construir un arreglo de enteros $[1, 2, ...N]$, con N igual al número total de ejemplos del texto a dividir. Cada elemento del arreglo representa el índice de un ejemplo en el conjunto.
2. Ordenar de forma aleatoria el arreglo. Este paso se ejecutó utilizando la función *shuffle* del módulo *random* de Python.
3. Iterar sobre el arreglo desde el inicio hasta el final. Para cada entero i perteneciente al arreglo, incluir el ejemplo i del texto dentro del nuevo conjunto de entrenamiento hasta llegar a iterar sobre el 70 % del largo del arreglo. Una vez sobrepasado el 70 %, continuar con la recorrida pero incluyendo los ejemplos restantes en el conjunto de evaluación.

Es interesante hacer algunas aclaraciones sobre este paso. El ordenamiento de las líneas se pierde: tomamos la decisión de seleccionar ejemplos de forma aleatoria para así obtener una distribución heterogénea de ejemplos dentro del corpus. La justificación de esta decisión subyace en el hecho de que en una gran proporción de documentos, la calidad de la imagen y por lo tanto del texto extraído es compartida a lo largo de todo el documento. Entonces si se tiene un documento de mala calidad, es probable que se tengan varias líneas de texto con un nivel de ruido alto, y en sentido contrario, si se tiene un documento claro y fácilmente legible, se contarán con líneas de texto de alta calidad. Bajo estas condiciones, tomar ejemplos de forma aleatoria nos parece un criterio más representativo del corpus comparado con seleccionar ejemplos ordenadamente dando la posibilidad de enviar un grupo de líneas buenas de texto hacia un conjunto y un grupo de líneas malas de texto hacia otro.

El número de ejemplos asignado al subconjunto de entrenamiento y al subconjunto de evaluación para ambos corpus (conjunto-O y conjunto-R) puede verse en la Tabla 6.17.

	Conjunto-O	Conjunto-R
Entrenamiento	4249	12213
Evaluación	1821	5234

Tabla 6.17: Número de ejemplos asignados para entrenamiento y evaluación

3) Alineación de palabras

Se alinean los datos con un nivel de granularidad de palabra utilizando GIZA++. Recordemos que los datos ya se encontraban alineados a nivel de línea u oración previo a este paso. Luego de ejecutar GIZA++ contaremos con el correspondiente de cada traducción a nivel de palabra, o sea, cada palabra de una oración origen tendrá su(s) palabra(s) asociada(s) en la oración destino. Hablamos en plural pues una palabra en una oración de un ejemplo puede tener más de una palabra asociada a ella en la otra oración correspondiente dentro del mismo ejemplo. Esto ocurre por errores tanto del lado del OCR como del lado de las reconstrucciones de LUISA.

Veamos un caso ocurrido en el entrenamiento para que sea más claro. Se tiene la oración del idioma origen (generada por el OCR):

Comisión do / ¿ licación (1)

Y su oración asociada:

Comisión de Aplicación (2)

En la oración 1 el OCR reconoció cinco palabras: “Comisión”, “do”, “/”, “¿” y “licación”. Mientras que en la oración 2 contamos con tres palabras únicamente: “Comisión”, “de” y “Aplicación”. Claramente el OCR se equivocó, confundió la letra “A” mayúscula con la barra inclinada “/”y la letra “p” minúscula con el signo de interrogación de apertura “¿”. No viene al caso, pero si analizamos la forma de los caracteres estos errores no son alocados, tienen su cuota de sentido. Volviendo a la alineación, este es un claro ejemplo donde una palabra debe tener a varias asociadas a ella. En la Tabla 6.18 podemos ver como terminaron siendo relacionadas entre ellas.

Token Oración 1	Token Oración 2
Comisión	Comisión
do	de
/	Aplicación
¿	Aplicación
licación	Aplicación

Tabla 6.18: Ejemplo de Alineación a nivel de palabra

El componente de alineación hizo un buen trabajo asociando los caracteres erróneos a la palabra correcta. Una vez finalizado el proceso de alineación se tiene este tipo de datos para cada pareja de oraciones.

4) Construcción de tablas de traducción léxica

Se toman las palabras alineadas del proceso anterior y se construyen dos tablas, una correspondiente a traducciones desde el idioma origen hacia el idioma destino, y otra tabla inversa de traducciones

destino-origen. Cada fila contiene tres elementos, una palabra en un idioma (ya sea origen o destino dependiendo de qué tabla hayamos seleccionado), una traducción asociada, y un número que representa la probabilidad dentro del corpus de traducir una palabra en otra. En el ejemplo de la Tabla 6.19 podemos ver las traducciones para la palabra "Carretas". A la probabilidad de traducir una palabra desde el idioma origen al idioma destino la nombraremos como $w(o|d)$, mientras que a la probabilidad de traducir palabras en sentido contrario, la simbolizaremos de forma general como $w(d|o)$.

Origen	Destino	$w(o\|d)$
Carretas	Carretas	0.7500000
etas	Carretas	0.2500000

Tabla 6.19: Tabla de traducción léxica origen-destino para la palabra "Carretas"

5) Extracción de frases

Todas las frases se vuelcan en un único archivo y se hace uso de las salidas de la alineación de palabras ejecutada con GIZA++ en el paso 3. Cada línea de este nuevo archivo está conformada por una frase en el idioma origen, su correspondiente traducción en el idioma destino y puntos de alineación. Los puntos de alineación son parejas de enteros (x,y) donde x representa la posición de una palabra en el idioma origen e y es la posición de una palabra en el idioma destino asociada a la palabra indicada por x. En la Tabla 6.20 podemos ver un extracto de una línea del archivo generado en este paso.

Frase Origen	Frase Destino	Ptos. de Alineación
csntinOsn o dispaaición da la auperiurided ,	continúan a disposición de la superioridad ,	0-0 1-1 2-2 3-3 4-4 5-5 6-6

Tabla 6.20: Ejemplo de extracción de frases

En este punto también se crea un archivo de extracciones de orden inverso, es decir, la primer columna de cada línea contiene una frase en el idioma destino (en lugar del idioma origen) y asociada a esta frase se encuentra su traducción en el idioma origen.

6) Puntuación de frases

A partir de las parejas de frases generadas por el proceso de extracción del paso 5 se crea una tabla de traducción desde el idioma origen al idioma destino. Este paso se encuentra separado del anterior simplemente por temas de performance y recursos.

En primera instancia, para cada fila del archivo de extracciones se calcula la probabilidad $\phi(o|d)$ de que la frase o del idioma origen sea la traducción de la frase d del idioma destino. A esta probabilidad la llamaremos probabilidad de traducción de frase directa. Esto se hace a través del conteo de ocurrencias de frases en ambos idiomas. Este proceso se repite utilizando el archivo de extracciones de orden inverso, obteniendo así las probabilidades de traducción $\phi(d|o)$ desde el idioma destino al idioma origen para cada pareja de frases. Esta probabilidad se conoce dentro del sistema como probabilidad de traducción de frase inversa.

A modo de resumen, para cada pareja de frases se calculan 2 valores principales (adicionalmente se pueden calcular otros valores de menor relevancia según distintas opciones definidas a la hora de invocar los procesos):

- Probabilidad de traducción de frase inversa $\phi(d|o)$
- Probabilidad de traducción de frase directa $\phi(o|d)$

Veamos que ocurre con las probabilidades tomando como ejemplo (desde la tabla generada en el entrenamiento del traductor para el conjunto-R) las traducciones de la frase *"Departamental de"* del idioma destino. En este caso la frase de ejemplo está conformada por dos palabras, pero a lo largo de la tabla de traducción la misma se irá incrementando con sus palabras adyacentes en el texto, calculándose las puntuaciones para cada una. Como ejemplos de frases que contienen a la seleccionada encontramos: *"Escuela Departamental de"*, *"la Escuela Departamental de" "la Escuela Departamental de Policía"*.

En la Tabla 6.21 vemos todas las traducciones asociadas al ejemplo elegido. Que el valor de $\phi(d|o)$ sea siempre 1 indica que cualquiera de las 5 frases de origen tiene una probabilidad de 1 de ser traducidas a la frase destino. En la práctica se llegó a este número porque toda oración del texto origen que contenía cualquiera de las frases listadas en la primer columna fue alineada con una oración del texto destino que contiene la frase *"Departamental de"*, y a su vez estas frases dentro de sus respectivas oraciones fueron alineadas entre sí a nivel de palabra.

Por otro lado, las probabilidades de traducciones $\phi(o|d)$ varían, pero denotando una frecuencia mayor cuando se trata de las frases iguales *"Departamental de"* en ambos idiomas, o en otras palabras, la mayoría de las veces en que se lee *"Departamental de"* en el texto destino, coincide con esa misma frase en el texto origen.

Frase Origen	Frase Destino	$\phi(o\|d)$	$\phi(d\|o)$
Departamental de	Departamental de	0.636364	1
Oopattaaenial de	Departamental de	0.0909091	1
D ˆ artamsntal de	Departamental de	0.0909091	1
ˆ artamsntal de	Departamental de	0.0909091	1
artamsntal de	Departamental de	0.0909091	1

Tabla 6.21: Ejemplo de puntuación de la frase "Departamental de"

7) Construcción de modelo de reordenamiento (o distorsión)

En este paso se construye un modelo de reordenamiento (también conocido como modelo de distorsión) que busca lidiar con el problema de reordenamiento al que se enfrentan los SMT. Este problema encuentra su origen en el hecho de que no todas las palabras en una oración pueden ser traducidas de forma consecutiva al idioma destino. Esto significa que existen palabras que deben ser salteadas y traducidas en un orden distinto al que tienen en el idioma origen.

Es así que estos modelos buscan asignar costos a los posibles reordenamientos que deban hacerse a la hora de traducir frases desde un idioma a otro. En particular, el modelo utilizado en este sistema otorga un costo lineal a la distancia de reordenamiento. Por ejemplo, saltearse dos palabras al momento de traducir costará dos veces el valor que cuesta saltearse una única palabra.

Entendemos que para el contexto en el que se entrenó el SMT, esta característica y el problema de reordenamiento en general no presentan graves inconvenientes ya que estamos traduciendo desde español a español, por lo que el orden de las palabras entre ambos textos es en la mayoría de los casos exactamente el mismo.

8) Construcción de archivo de configuración

Este es el paso final del *pipeline*, en el mismo se genera un archivo de configuración que contiene metadatos a ser utilizados por el **decodificador**.

En este archivo encontraremos principalmente rutas hacia los distintos componentes generados durante el entrenamiento más un conjunto de parámetros de configuración.

6.2.4. Decodificador

El decodificador es el otro componente principal del SMT. Como su nombre indica, es quien se encarga de la etapa de decodificación, en la cual se intenta encontrar la traducción más probable para una frase dada en el idioma origen utilizando un modelo lineal. Este modelo queda definido a través de distintas características generadas durante la etapa previa de entrenamiento, entre las que encontramos: probabilidades calculadas para las tablas de frases y reglas, probabilidades del modelo de lenguaje, modelo de reordenamiento, junto con conteos a nivel de palabra y frase.

Este proceso trabaja con el concepto de *opción de traducción* desde un inicio, por eso pasaremos a definirlo de manera más formal.

Opción de traducción

Dada una secuencia de palabras a modo de entrada, esta puede traducirse en cierto número de frases, que por lo general es significativamente mayor a uno. Cada una de estas frases se denomina *opción de traducción*. Veamos un ejemplo ilustrativo de esta situación en la Imagen 6.3 utilizada en la documentación web de Moses. En esta se listan las distintas opciones de traducción para la frase de entrada "María no daba una bofetada a la bruja verde".

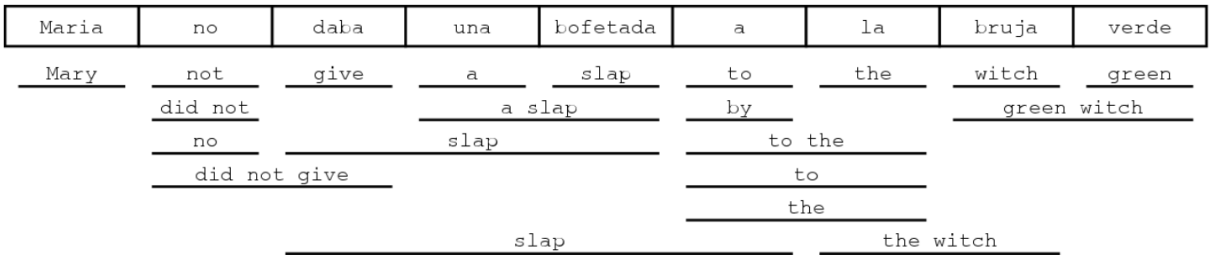


Imagen 6.3: Ejemplo de opciones de traducción

Las opciones de traducción asociadas a una frase que quiera traducirse son calculadas previo a que la decodificación en sí tome lugar. Estas son consideradas como punto de partida a la hora de buscar la mejor traducción, esto es, se trabajará con subsecciones de la tabla de frases que contengan entera o parcialmente a las opciones de traducción en lugar de realizar búsquedas sobre toda la tabla (la cual es inevitablemente de gran tamaño).

Algoritmo principal

El decodificador de Moses utiliza el algoritmo "Beam Search" para llevar a cabo la búsqueda de la mejor traducción. "Beam Search" es un algoritmo de búsqueda basado en la exploración de un grafo, expandiendo los nodos más prometedores de un conjunto limitado de opciones según alguna métrica o puntaje dados. En este encontraremos como regla general que las frases de salida son generadas de izquierda a derecha en forma de hipótesis.

La búsqueda comienza en un estado inicial en donde no se ha seleccionado ninguna palabra de la frase origen para traducir y por lo tanto no se ha generado ninguna traducción. Los nuevos estados

son creados cuando la salida se ve extendida debido a que se consideraron nuevas palabras de la frase origen.

Siguiendo con el ejemplo visto anteriormente, en la Imagen 6.4 apreciamos el grafo construido tras las primeras iteraciones del algoritmo. El nodo de más a la izquierda es el nodo inicial al que se hizo referencia anteriormente. En este esquema reducido de los estados, se detallan 3 atributos para cada uno: la frase resultante en ese paso (e), las palabras traducidas de la frase original (f) y la probabilidad asociada al fragmento traducido (p).

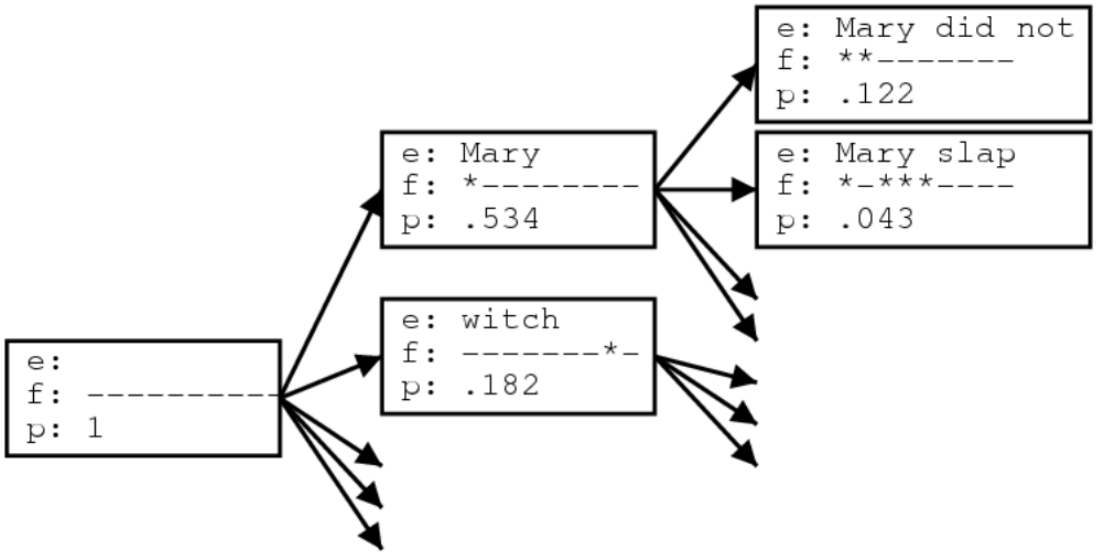


Imagen 6.4: Ejemplo del algoritmo “Beam Search”

A medida que el algoritmo avanza, generará nuevos estados a partir de los anteriores. El costo de un nuevo estado es igual al costo del estado previo multiplicado por cuatro costos probabilísticos adicionales asociados a las nuevas traducciones parciales tomadas en cuenta. Estos cuatro factores son obtenidos a partir de los siguientes componentes:

- **Tabla de traducción de frases**: Busca asegurar que las traducciones son acertadas.
- **Modelo de lenguaje**: Principalmente trata de garantizar fluidez en el idioma objetivo.
- **Modelo de distorsión**: Permite el reordenamiento de palabras, asociando costos a cada cambio.
- **Penalidad de palabra o “word penalty”**: Penaliza traducciones demasiado cortas o demasiado largas.

Los estados finales en la búsqueda son alcanzados cuando se cubren todas las palabras del idioma origen. Dentro de estos, se selecciona la hipótesis con el costo más bajo, o dicho de otra manera, se selecciona la hipótesis con la probabilidad más alta. Es esta hipótesis quien finalmente es devuelta como traducción de la frase original.

Este algoritmo descrito a grandes rasgos puede utilizarse de forma exhaustiva, aunque esto no es conveniente debido a la enorme cantidad de hipótesis posibles a evaluar. Por esta razón es que en la práctica se aplican una serie de estrategias con el fin de optimizar la búsqueda. Las optimizaciones vienen dadas por el lado de las podas y el descarte de hipótesis intermedias las cuales se estima son de mucho mayor costo que otras hipótesis con las que se esté trabajando.

Evaluación

El desempeño de los dos sistemas de traducción entrenados fue evaluado utilizando el puntaje BLEU. No debimos implementar ningún paso extra para esto ya que el *toolkit* de Moses cuenta con un módulo de evaluación integrado que utiliza esta medida. El puntaje fue calculado sobre los conjuntos de evaluación construidos para cada enfoque. Recordemos que el rango de valores que puede tomar BLEU comprende los racionales entre 0 y 1 (incluyendo a los extremos), siendo 1 el mejor puntaje posible y representando una traducción perfecta.

Es así que el sistema de traducción automática construido a partir del enfoque de alineación por oraciones obtuvo un puntaje BLEU de **0.3332** mientras que el traductor que utilizó el enfoque alternativo de alineación por renglones logró un puntaje BLEU de **0.4211**. Finalmente resultó vencedor el enfoque de alineación por renglones. El desempeño del traductor construido a partir de este último enfoque será comparado más adelante contra la mejor configuración de la solución basada en modelo de lenguaje.

Si bien no es recomendado comparar puntajes BLEU obtenidos por sistemas de traducción entrenados para distintos lenguajes y sobre distintos corpus paralelos, creemos que ver algunos ejemplos de puntajes alcanzados en otras investigaciones puede ayudar a la valorización en cierta medida de los resultados aquí obtenidos.

Refiriéndonos al la investigación en la que se define el modelo con el que trabaja Moses [52] podemos ver que se presentan distintos experimentos y se compara la performance de más de un sistema de traducción. En este trabajo los puntajes BLEU varían normalmente en el rango 0.20 a 0.28 y cabe resaltar que la idea del mismo es entrenar un sistema de traducción convencional y no utilizarlo como corrector.

Por otro lado Afli et al. [53] así como nosotros en este proyecto exploraron el uso de traductores automáticos para la corrección de textos generados por sistemas OCR y evaluaron los resultados a través del puntaje BLEU. La particularidad de este proyecto está en la buena calidad de los textos iniciales producidos por el OCR, estos contaron con puntajes BLEU iguales o mayores a 0.9 antes de haber realizado cualquier tipo de corrección. De todas formas, los traductores entrenados lograron mejorar los textos, incrementando de esta manera los puntajes. En promedio se vieron mejoras de entre 0.02 a 0.07 sobre los números obtenidos por el OCR.

Por último, nos gustaría compartir una tabla comparativa (tabla 6.22) de puntajes BLEU presentada en la documentación del producto Cloud AutoML [54] de Google. Allí se aclara que es una interpretación que puede servir como guía de manera muy general.

Puntuación BLEU	Interpretación
< 0.10	Casi inútil
0.10 - 0.19	Difícil de captar la esencia
0.20 - 0.29	La esencia es clara, pero tiene errores gramaticales significativos
0.30 - 0.39	Traducciones comprensibles a buenas
0.40 - 0.49	Traducciones de alta calidad
0.50 - 0.60	Traducciones de calidad muy alta, adecuadas y fluidas
> 60	Calidad que suele ser mejor que la humana

Tabla 6.22: Interpretación general de puntajes BLEU

Capítulo 7

Resultados

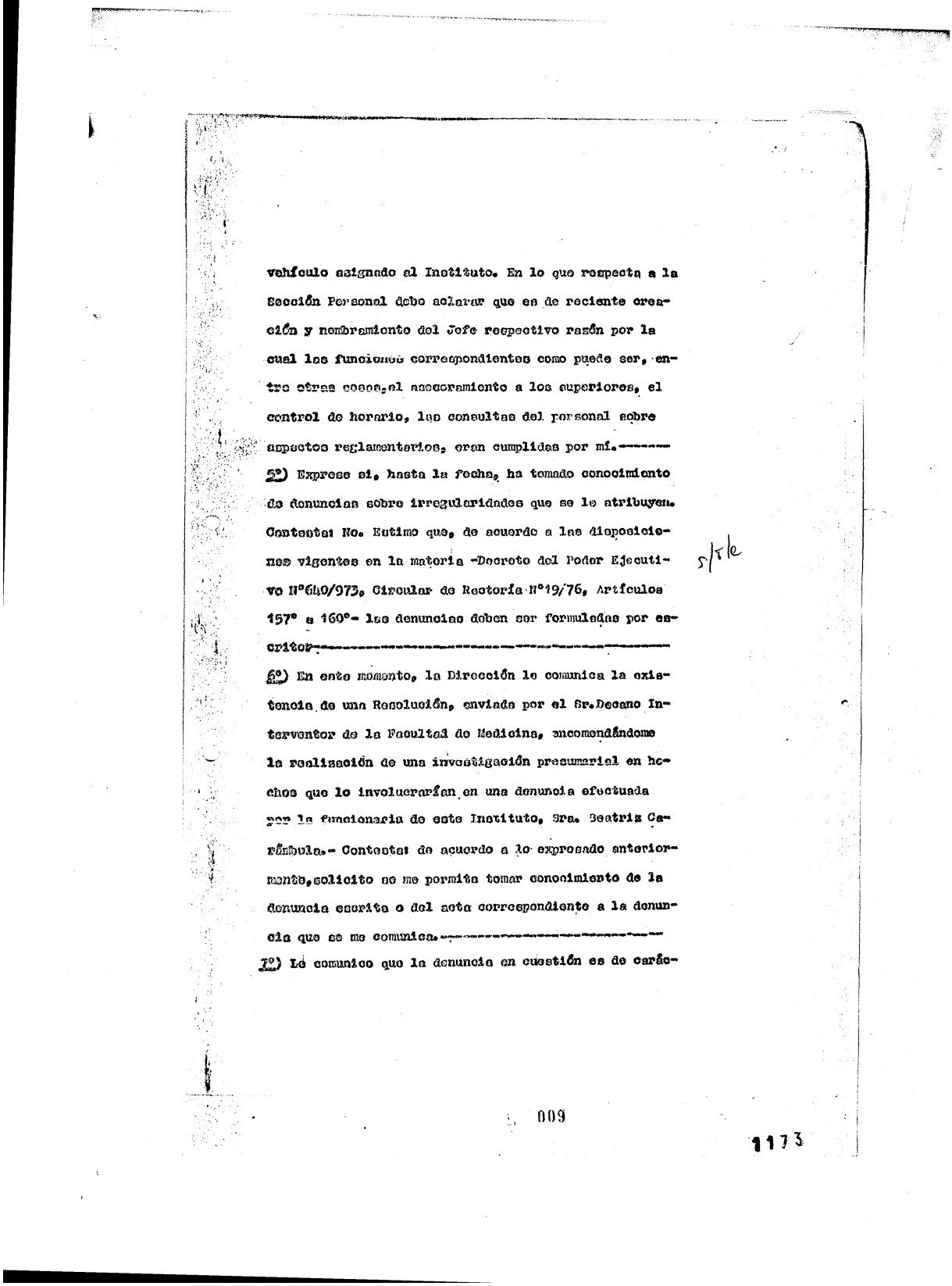
Para presentar los resultados del proyecto, se decidió elegir un documento específico y corregirlo utilizando ambas estrategias, mostrando y comparando los detalles de cada una de las correcciones. En el caso del modelo de n-gramas, se corrige utilizando la mejor configuración obtenida en la prueba de la sección 5.3.1, y por otro lado, para el modelo de traducción automática se hizo utilizando el enfoque entrenado por líneas. También se calculó para 544 documentos, de los cuales se contaba con su transcripción reconstruida a partir de LUISA, el puntaje promedio de similitud usando SequenceMatcher, comparando el puntaje promedio antes de la corrección y después. Hay que destacar que entre los 544 documentos, no se encuentra ninguno de los que se utilizó para el entrenamiento del sistema SMT, por lo que la evaluación se realiza sobre un universo no explorado.

7.1. Corrección detallada

El documento escogido para transcribir y corregir los errores cometidos por el OCR al transcribir, se muestra en la Imagen 7.1. Cabe destacar que el OCR utilizado (Abbyy Finereader) reconoce perfectamente el texto del documento e ignora todos los bordes y manchas que no resultan importantes, esto se puede ver en la Imagen 7.2. La salida original obtenida por el OCR del documento presenta en total 57 errores corregibles (palabras coloreadas en la Tabla 7.1 excepto las 2 detectadas como error incorrectamente) y tiene un puntaje de similitud de 0.871 medido con SequenceMatcher.

7.1.1. Usando modelo de n-gramas

En la Tabla 7.1 se muestran: del lado izquierdo la salida original de procesar la Imagen 7.1 con Abbyy Finereader, y del lado derecho el resultado de corregir utilizando la estrategia con el modelo de n-gramas con la mejor configuración obtenida en la prueba de la sección 5.3.1. Se destacan con distintos colores tanto los errores detectados y corregidos correctamente, errores no detectados, detecciones de error incorrectamente y errores corregidos incorrectamente. También se distinguen palabras que el OCR transcribió incorrectamente pero no contienen errores ortográficos.



vehículo asignado al Instituto. En lo que respecta a la Sección Personal debo aclarar que es de reciente creación y nombramiento del Jefe respectivo razón por la cual las funciones correspondientes como puede ser, entre otras cosas, el asesoramiento a los superiores, el control de horario, las consultas del personal sobre aspectos reglamentarios, eran cumplidas por mí.-------

2º) Exprese si, hasta la fecha, ha tomado conocimiento de denuncias sobre irregularidades que se le atribuyen. Contesta: No. Estimo que, de acuerdo a las disposiciones vigentes en la materia -Decreto del Poder Ejecutivo Nº640/973, Circular de Rectoría Nº19/76, Artículos 157º a 160º- las denuncias deben ser formuladas por escrito.----------

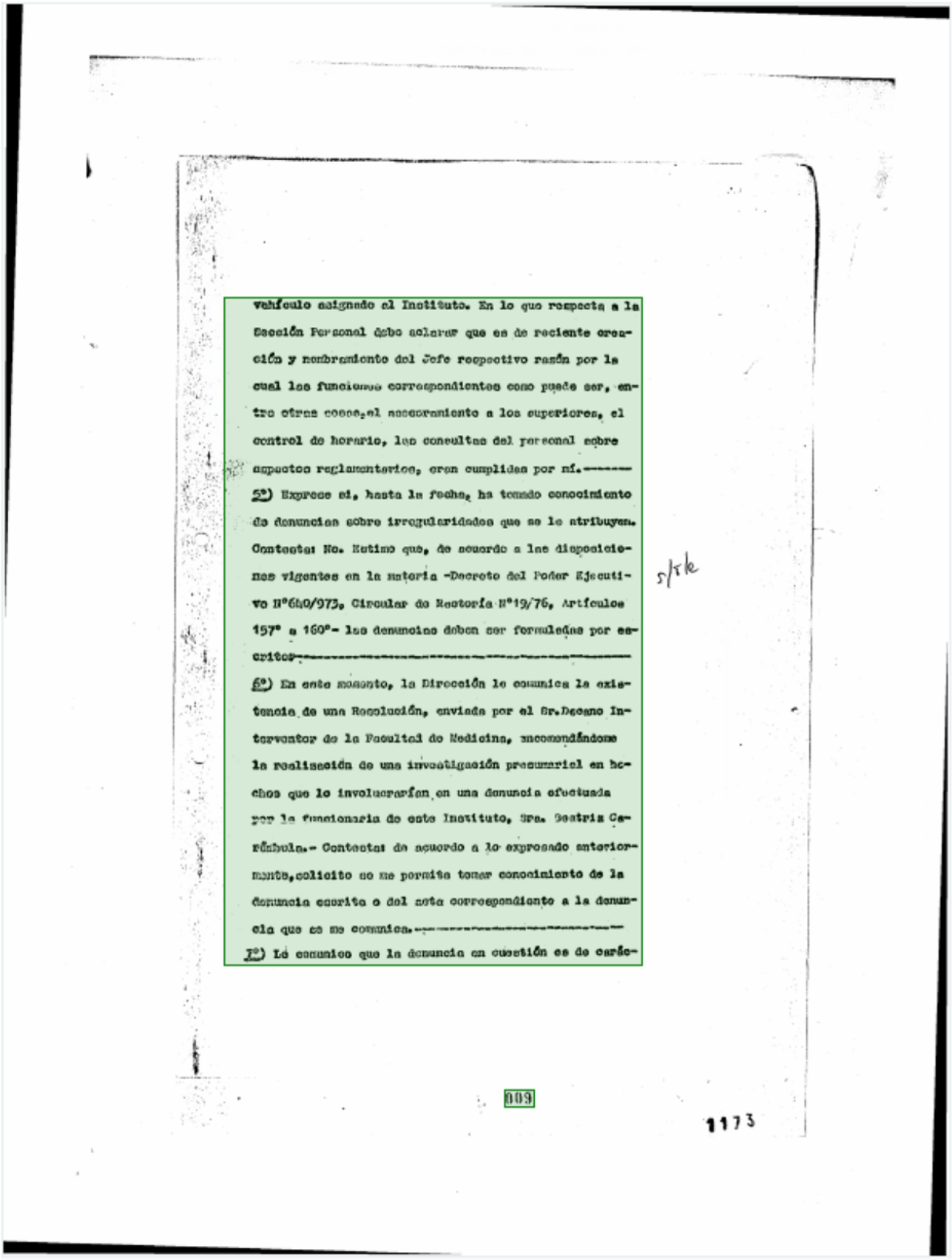
3º) En este momento, la Dirección le comunica la existencia de una Resolución, enviada por el Sr.Decano Interventor de la Facultad de Medicina, encomendándome la realización de una investigación presumarial en hechos que lo involucrarían en una denuncia efectuada por la funcionaria de este Instituto, Sra. Beatriz Carámbula.- Contesta: de acuerdo a lo expresado anteriormente, solicito se me permita tomar conocimiento de la denuncia escrita o del acta correspondiente a la denuncia que se me comunica.----------

4º) Le comunico que la denuncia en cuestión es de carác-

009

1173

Imagen 7.1: Documento a transcribir y corregir.



vehículo asignado al Instituto. En lo que respecta a la Sección Personal debo aclarar que es de reciente creación y nombramiento del Jefe respectivo razón por la cual las funciones correspondientes como puede ser, entre otras cosas,el acercamiento a los superiores, el control de horario, las consultas del personal sobre aspectos reglamentarios, eran cumplidas por mí.------

5º) Expresa si, hasta la fecha, ha tenido conocimiento de denuncias sobre irregularidades que se le atribuyen. Contesta: No. Estimo que, de acuerdo a las disposiciones vigentes en la materia -Decreto del Poder Ejecutivo Nº640/973, Circular de Rectoría Nº19/76, Artículos 157º a 160º- las denuncias deben ser formuladas por escrito.----------

6º) En este momento, la Dirección le comunica la existencia de una Resolución, enviada por el Sr.Decano Interventor de la Facultad de Medicina, recomendándose la realización de una investigación preumarial en hechos que lo involucrarían en una denuncia efectuada por la funcionaria de este Instituto, Sra. Beatriz Carámbula.- Contesta: de acuerdo a lo expresado anteriormente,solicito se me permita tomar conocimiento de la denuncia escrita o del acta correspondiente a la denuncia que se me comunica.----------

7º) Le comunico que la denuncia en cuestión es de carác-

009

1173

Imagen 7.2: Reconocimiento de texto en el documento realizado por Abbyy.

Transcripción Abbyy original	Trancripción corregida modelo n-gramas
vehículo asignado al Instituto* En lo qua rospectq a la Sección Personal debo aclara? que es de reciente creación y non&ramionto doi Jefe respectivo rasÓn por la cual las funcitmoM correapondienteo como puede ser, entre otras coop$-el asesoramientc a los superiores, el control de horario, leo consultas del personal sobre s aspectos regí arcan!; avíos- oran cumplidas por mí*——— Expreso si, hasta la focha, ha tomado conocimiento do denuncias sobre irrcgulítridades que se le atribuyen* Contesta! No. Estima que, de acuerdo a las di oposiciones vigentes en la materia -Decreto del Poder Ejecutivo N°6h0/973* Circular de Rectoría N"19/'76, Artículos 157^ a 160^- leo denuncios deben ser formuladas por escrltoy-———————————————— —- En cate momento, la Dirección lo comunica la oxistenoia de una Resolución, enviada por al Sr.Deoeno Interventor de la Facultad do Medicina, encomendándome la raaliaaoidn de una investigación presumariel en hechos que lo involucrarían en una denuncia ef oatoada po? 15 funcionarla de este Instituto, 3?a* Beatriz Ca— rÓEibulR*- Contoata! do acuerdo a lo exprosado anteriornwitb, solicito so me pormita tomar conocimiento de la denuncia escrita o del acta correspondiente a la denuncia que ce me comunica*–'————-' ——— Lo esmunlco que la denuncia en cuestión es de car&c-009	vehículo asignado al Instituto* En lo que respecta a la Sección Personal debo aclara? que es de reciente creación y nombramiento dio Jefe respectivo razón por la cual las funciono correspondiente como puede ser, entre otras cook$el asesoramiento a los superiores, el control de horario, leo consultas del personal sobre s aspectos regí marcan!; avíos oran cumplidas por mí*——— Expreso si, hasta la fecha, ha tomado conocimiento de denuncias sobre irregularidades que se le atribuyen* Contesta! No. Estima que, de acuerdo a las di oposiciones vigentes en la materia decreto del Poder Ejecutivo N°6h0/973* Circular de Rectoría N"19/'76, Artículos 157^a 160^- leo denuncios deben ser formuladas por esa crltoy-———————————————— —- En cate momento, la Dirección lo comunica la existencia de una Resolución, enviada por al Sr.Deoeno Interventor de la Facultad de Medicina, encomendando la realización de una investigación sumarial en hechos que lo involucrarían en una denuncia ef atoada por? 15 funcionaría de este Instituto, 3?a* Beatriz Ca— rÓEibulR*- Contoata! de acuerdo a lo expresado anteriormente, solicito se me permita tomar conocimiento de la denuncia escrita o del acta correspondiente a la denuncia que se me comunica*–'————-'——— Lo esculco que la denuncia en cuestión es de car&c-009

Tabla 7.1: Comparación salida original Abbyy contra su corrección con n-grams

Error corregido correctamente.
Error corregido incorrectamente.
Error no detectado.
Error detectado incorrectamente.
Ortografía correcta, error en la transcripción no detectado.

Para llegar al resultado que se observa en el lado derecho de la tabla, se realiza un procesamiento en dos partes:

1. Se verifica si las palabras que están cortadas por un guion y salto de línea, al juntarlas (eliminando el guion) componen una palabra sin errores ortográficos, en caso negativo se corrigen. Se marcan las palabras sin errores para que sean ignoradas en el siguiente paso.
2. Se busca una corrección para cada una de las palabras con errores ortográficos excepto las marcadas en el paso anterior.

Las palabras observadas y la corrección (o no) del paso 1 para el documento se muestran en la Tabla 7.2. De los 11 casos en total, se encuentran:

- 6 ($\approx 55\,\%$) palabras que no tienen error y no hace falta corregir.
- 2 ($\approx 18\,\%$) palabras con errores ortográficos y que se corrigieron bien.
- 1 ($\approx 9\,\%$) palabra que no tiene error ortográfico pero no es correcta en la transcripción.
- 2 ($\approx 18\,\%$) palabras que tenían errores ortográficos y no se corrigieron.

Las 2 palabras que no se corrigieron, son las únicas palabras que no son marcadas para ser ignoradas en el paso 2, con el objetivo de buscar una corrección en este segundo paso. Particularmente, la palabra

errónea “es- crltoy–” no se corrigió a causa de los guiones posteriores en el documento, dado que dos de esos guiones se tomaron como parte del token y al buscar una corrección, no se encontró ninguna que se encuentre a una distancia chica respecto de la misma. Viendo la imagen del documento y el contexto se puede identificar que en el caso de la palabra “car&c- 009” la palabra correcta es “carácter”, pero es un caso muy difícil para un método de corrección automático, ya que falta la otra mitad de la palabra debido a que se encuentra en la siguiente hoja, y que además el OCR identificó y transcribió también lo que es probablemente el número de página.

Este primer paso para marcar y/o corregir es importante (aunque no es perfecto y se puede mejorar) ya que si no se hiciera, todas las 11 palabras serían encontradas como error en el segundo paso, dado que al tokenizar se obtienen dos tokens distintos para cada palabra separada, y posiblemente se detectaría como error y corregiría cada uno de esos tokens.

Transcripción Abbyy original	Corrección paso 1	Error ortográfico	Correcto
crea- ción	crea- ción	No	Sí
en- tre	en- tre	No	Sí
oposicio- nes	oposicio- nes	No	No
Ejecuti- vo	Ejecuti- vo	No	Sí
es- crltoy--	es- crltoy--	Sí	No
oxis- tenoia	exis- tencia	Sí	Sí
In- terventor	In- terventor	No	Sí
he- chos	he- chos	No	Sí
anterior- nwitb	anterior- mente	Sí	Sí
denun- cia	denun- cia	No	Sí
car&c- 009	car&c- 009	Sí	No

Tabla 7.2: Corrección y marcado palabras cortadas por guion y salto de línea

Los detalles de cada palabra observada en el segundo paso se encuentran en la Tabla 10.6 del anexo, mostrando para cada palabra identificada como error: el contexto del lado izquierdo, el contexto del lado derecho, las sugerencias de corrección, la distancia de Damerau–Levenshtein entre cada sugerencia y el error, la frecuencia de aparición de las sugerencias junto a los contextos (brindada por PhraseFinder) y la corrección del error.

De los 57 errores en total, se obtuvieron los siguientes resultados luego del procesamiento:

- 20 ($\approx 35\,\%$) errores corregidos correctamente.
- 12 ($\approx 21\,\%$) errores corregidos incorrectamente.
- 18 ($\approx 32\,\%$) errores no detectados. Incluyendo 7 signos de puntuación, 1 letra suelta, 2 palabras con signos y 8 palabras alfanuméricas.
- 7 ($\approx 12\,\%$) errores de transcripción no detectados, siendo palabras correctas.

Además:

- 2 errores detectados incorrectamente.
- Puntaje de similitud luego de la corrección: 0.891

El resultado de la corrección para este documento es aceptable considerando que gran parte de los errores fueron corregidos correctamente. En este caso se puede ser incluso mucho más optimista y

no considerar dentro del total de errores a 12 de los 18 errores no detectados, por tratarse de signos de puntuación aislados y números de referencia que no agregan demasiado valor al corregirlos. Sin contar estos 12 errores, la cantidad total sería 45 y por lo tanto el porcentaje de errores corregidos correctamente aumentaría a 44 % aproximadamente.
Por último, el valor del puntaje de similitud se incrementó 0.02, que significa un aumento realmente bueno como se mostró en la prueba realizada para comparar las configuraciones en 5.3.1

7.1.2. Usando sistema de traducción automática estadística

En el caso de la corrección usando el sistema de traducción automática estadística Moses, los resultados difieren considerablemente al enfoque anterior. Los errores resaltados y la corrección obtenida por este sistema se muestran en la Tabla 7.3. A simple vista es muy claro que los resultados obtenidos son peores que usando el modelo de n-gramas.

Este enfoque no verifica errores a nivel de ortografía, por lo tanto en este caso no diferenciamos las palabras con ortografía correcta que son un error en la transcripción. Por otro lado, se introduce un caso nuevo ("error introducido" destacado en celeste) que es: la introducción de palabras no existentes a la transcripción.

Transcripción Abbyy original	Trancripción corregida SMT
vehículo asignado al Instituto* En lo qua rospectq a la Sección Personal debo aclara? que es de reciente creación y non&ramionto doi Jefe respectivo rasÓn por la cual las funcitmoM correapondienteo como puede ser, entre otras coop$-el asesoramientc a los superiores, el control de horario, leo consultas del personal sobre s aspectos regí arcan!; avíos- oran cumplidas por mí*——— Expreso si, hasta la focha, ha tomado conocimiento do denuncias sobre irrcgulítridades que se le atribuyen* Contesta! No. Estima que, de acuerdo a las di oposiciones vigentes en la materia -Decreto del Poder Ejecutivo N°6h0/973* Circular de Rectoría N"19/'76, Artículos 157^ a 160^- leo denuncios deben ser formuladas por escrltoy-———————————————— —- En cate momento, la Dirección lo comunica la oxistenoia de una Resolución, enviada por al Sr.Deoeno Interventor de la Facultad do Medicina, encomendándome la raaliaaoidn de una investigación presumariel en hechos que lo involucrarían en una denuncia ef oatoada po? 15 funcionarla de este Instituto, 3?a* Beatriz Ca— rÓEibulR*- Contoata! do acuerdo a lo exprosado anteriornwitb, solicito so me pormita tomar conocimiento de la denuncia escrita o del acta correspondiente a la denuncia que ce me comunica*–'————-' ——— Lo esmunlco que la denuncia en cuestión es de car&c- 009	ejem vehículo asignado al Instituto En lo que rospectq a la Sección Personal debe aclara que es de reciente creación y nos de ramionto del Jefe respectivo rasÓn por la cual las funcitmoM correapondienteo como puede ser , enENTRE otras coop $ del asesoramientc a los superiores , el control de horario , las consultas del personal sobre s aspectos regí arcan de avíos- eran cumplidas por mí . – – expreso si , hasta la fecha , ha tomado conocimiento de denuncias sobre irrcgulítridades que se le atribuyen . Contesta : no . estima que , de acuerdo a las di oposiciones vigentes en la materia -Decreto del Poder Ejecutivo N ° 6h0 / 973 . circular de Rectoría N ° 19 / Fórmua 05 . 9 / / artículos 157 a 160 . las denuncios deben ser formuladas por escrltoy- – ———— – – – – ———— – – —- en este momento , la Dirección la comunica la oxistenoia de una Resolución , enviada por el Sr.Deoeno In- . terventor de la Facultad de Medicina , encomendándome la raaliaaoidn de una investigación presumariel en hechos que lo involucrarían en una denuncia años oatoada po ? 15 funcionario de este Instituto , 3 la . Beatriz Carmen – rÓEibulR . Contoata : de acuerdo a lo exprosado anteriornwitb , solicito se me pormita tomar conocimiento de la denuncia escrito o del acta correspondiente a la denuncia que se me comunica . – — – – - – - lo esmunlco que la denuncia en cuestión es de ser de o 009

Tabla 7.3: Comparación salida original Abbyy contra su corrección con SMT

Error corregido correctamente.
Error corregido incorrectamente.
Error no detectado.
Error detectado incorrectamente.
Error introducido.

De los 57 errores totales, luego de la corrección se obtienen los siguientes resultados:

- 19 ($\approx 33\,\%$) errores corregidos correctamente. Incluyendo 8 signos de puntuación y 11 palabras.
- 16 ($\approx 28\,\%$) errores corregidos incorrectamente.
- 22 ($\approx 39\,\%$) errores no detectados.

Además:

- 4 errores detectados incorrectamente.
- 2 errores introducidos.
- Puntaje de similitud luego de la corrección: 0.845

Si se comparan solamente estos números con el caso anterior, no parece haber una gran diferencia, pero comparando la Tabla 7.1 y la Tabla 7.3 se puede ver que las palabras corregidas por el primer enfoque agregan más valor a la corrección, teniendo en cuenta que casi la mitad de los errores corregidos por el traductor fueron signos de puntuación, y además este último introduce nuevos errores. Comparado al caso anterior, si no se cuentan los signos de puntuación y los números de referencia incluidos en el documento, la cantidad total de errores nuevamente es 45 y el porcentaje de errores corregidos de 24 % aproximadamente, que además de tratarse de un decremento, es un porcentaje de corrección bajo.
El puntaje de similitud obtenido luego de la corrección disminuyó 0.026 su valor, reflejando que el documento se alejó de su mejor versión luego del procesamiento con el traductor, o lo que es lo mismo, que se introdujeron más diferencias que similitudes hacia documento objetivo.

Como ya se mencionó en el Capítulo 6, el entrenamiento del sistema no se hizo con una cantidad adecuada de ejemplos, por lo tanto son esperables los resultados obtenidos. Lo destacable de la corrección es que se puede observar que encontró patrones correctamente, que dan a entender que son errores de lectura por parte del OCR que se repiten en muchos otros documentos, como por ejemplo: “qua” lo corrigió como “que”, “cate” como “este”, “focha” como “fecha”, etc. Incluso para los signos de puntuación como el signo de exclamación (!), que lo corrigió correctamente por dos puntos (:) y eliminó los asteriscos y comillas aisladas.

7.2. Evaluación del modelo de n-gramas

Nuevamente, utilizando la mejor configuración obtenida para este enfoque, se realizó la corrección de 544 documentos nuevos, con la finalidad de comparar el promedio de similitud entre los documentos sin corregir y LUISA, contra la similitud entre los documentos corregidos y LUISA. Los resultados se muestran en la Tabla 7.4, observando un aumento de 0.007 en el puntaje promedio luego de la corrección.

	Antes de corregir	Después de corregir
Promedio puntaje	0.543	0.550

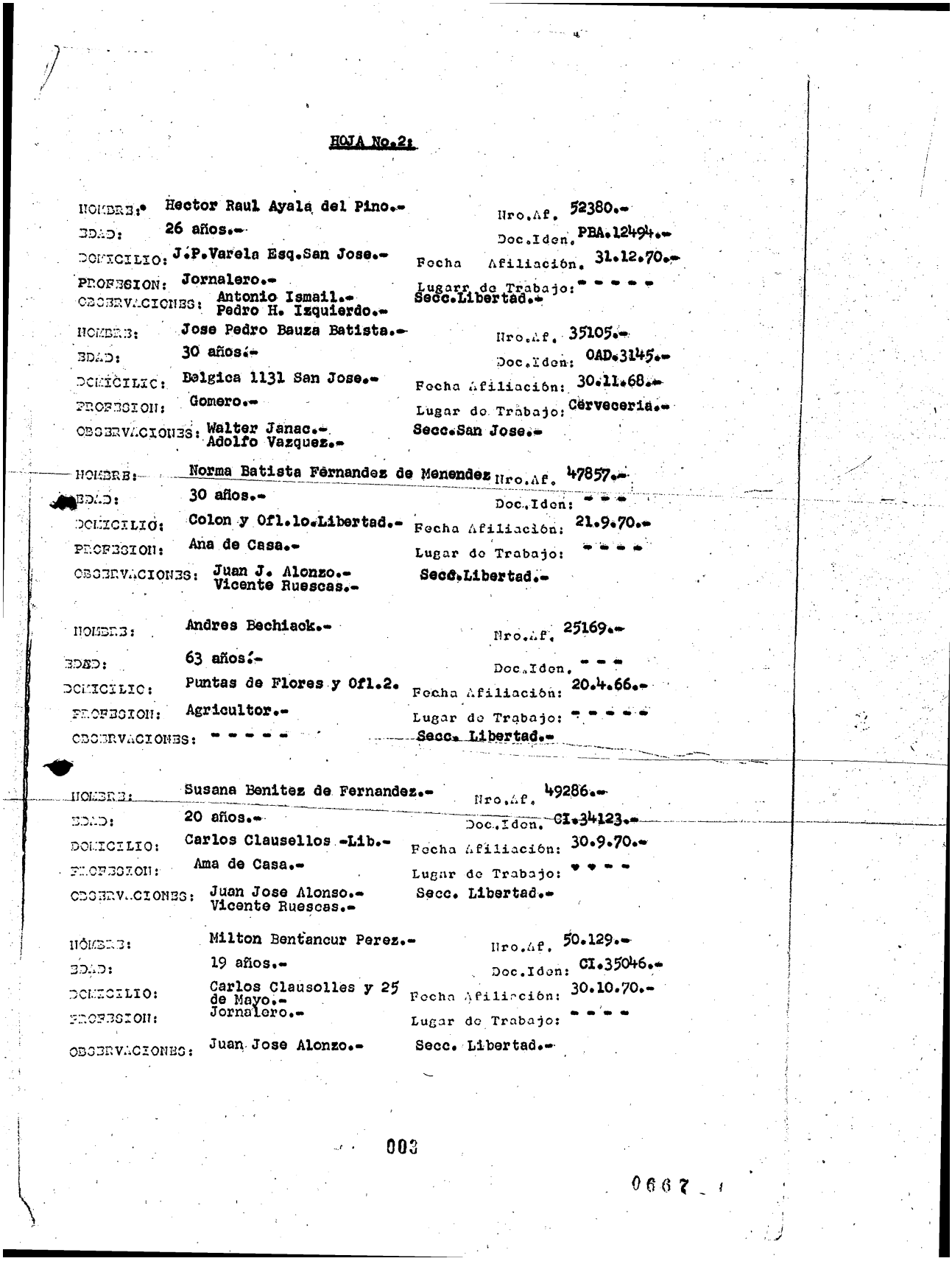
Tabla 7.4: Promedio puntaje de similitud antes y después de corregir 544 documentos usando el modelo de n-gramas.

También se observa en la Tabla 7.5, que 385 de los 544 documentos obtuvieron un incremento en el puntaje de similitud, lo que corresponde aproximadamente al 71 % del total de documentos.

	Mejora puntaje	Disminución puntaje
Cantidad de documentos	385	159

Tabla 7.5: Cantidad de documentos entre los 544 iniciales con mejora o disminución de puntaje de similitud luego de la corrección.

Sin olvidar que también son muchos los casos en los que no se logra una corrección positiva, una vez más, se observa que la corrección realizada obtiene resultados positivos. Los documentos para los cuales la corrección no es buena, generalmente son documentos que son de muy mala calidad, están estructurados como tablas, o directamente son solo figuras como huellas digitales o nombres propios. Un ejemplo es el que se muestra en la Imagen 7.3, un documento donde la mayoría de las palabras son nombres propios, y que tiene un puntaje de similitud inicial de 0.487, y luego de la corrección 0.433.



HOJA No.2:

NOMBRE: Hector Raul Ayala del Pino.- Nro.Af. 52380.-
EDAD: 26 años.- Doc.Iden. PBA.1249[illegible].-
DOMICILIO: J.P.Varela Esq.San Jose.- Fecha Afiliación. 31.12.70.-
PROFESION: Jornalero.- Lugarr de Trabajo: - - - - -
OBSERVACIONES: Antonio Ismail.- Pedro H. Izquierdo.- Secc.Libertad.-

NOMBRE: Jose Pedro Bauza Batista.- Nro.Af. 35105.-
EDAD: 30 años.- Doc.Iden: OAD.3145.-
DOMICILIO: Belgica 1131 San Jose.- Fecha Afiliación: 30.11.68.-
PROFESION: Gomero.- Lugar de Trabajo: Cerveceria.-
OBSERVACIONES: Walter Janac.- Adolfo Vazquez.- Secc.San Jose.-

NOMBRE: Norma Batista Fernandez de Menendez Nro.Af. 47857.-
EDAD: 30 años.- Doc.Iden: - - -
DOMICILIO: Colon y Ofl.lo.Libertad.- Fecha Afiliación: 21.9.70.-
PROFESION: Ama de Casa.- Lugar de Trabajo: - - - -
OBSERVACIONES: Juan J. Alonzo.- Vicente Ruescas.- Secc.Libertad.-

NOMBRE: Andres Bechiack.- Nro.Af. 25169.-
EDAD: 63 años.- Doc.Iden. - - -
DOMICILIO: Puntas de Flores y Ofl.2. Fecha Afiliación: 20.4.66.-
PROFESION: Agricultor.- Lugar de Trabajo: - - - - -
OBSERVACIONES: - - - - - Secc. Libertad.-

NOMBRE: Susana Benitez de Fernandez.- Nro.Af. 49286.-
EDAD: 20 años.- Doc.Iden. CI.34123.-
DOMICILIO: Carlos Clausellos -Lib.- Fecha Afiliación: 30.9.70.-
PROFESION: Ama de Casa.- Lugar de Trabajo: - - - -
OBSERVACIONES: Juan Jose Alonso.- Vicente Ruescas.- Secc. Libertad.-

NOMBRE: Milton Bentancur Perez.- Nro.Af. 50.129.-
EDAD: 19 años.- Doc.Iden: CI.35046.-
DOMICILIO: Carlos Clausolles y 25 de Mayo.- Fecha Afiliación: 30.10.70.-
PROFESION: Jornalero.- Lugar de Trabajo: - - - -
OBSERVACIONES: Juan Jose Alonzo.- Secc. Libertad.-

003

0667

Imagen 7.3: Documento de ejemplo disminución de puntaje.

7.3. Evaluación del modelo de traducción automática estadística

El promedio de puntaje de similitud antes y después de corregir que se obtiene para los mismos 544 documentos que la parte anterior, pero en este caso usando el modelo de traducción automática estadística, se muestran en la Tabla 7.6. En este caso vemos que el puntaje promedio decrece luego del procesamiento, lo que a grandes rasgos quiere decir que se introdujeron más errores de los que se corrigieron. En particular decrece 0.029 su valor, que según se observó en la corrección detallada y en 5.3.1 (cambios de 0.02 en el puntaje suponen cambios importantes) es un cambio muy grande en el puntaje de similitud. Teniendo en cuenta que es una disminución en el puntaje, el resultado es realmente negativo.

	Antes de corregir	Después de corregir
Promedio puntaje	0.543	0.514

Tabla 7.6: Promedio puntaje de similitud antes y después de corregir 544 documentos usando el modelo de traducción automática estadística.

La diferencia de promedio de puntajes también se ve reflejado en la cantidad de documentos que sufrieron una disminución del puntaje en comparación con los que lo incrementaron. La Tabla 7.7 refleja que aproximadamente el 24 % de los documentos obtuvieron un aumento en el puntaje.

	Mejora puntaje	Disminución puntaje
Cantidad de documentos	132	412

Tabla 7.7: Cantidad de documentos entre los 544 iniciales con mejora o disminución de puntaje de similitud luego de la corrección.

Capítulo 8

Conclusiones

Este proyecto se centró en la mejora de los procesos de análisis de documentos históricos, los cuales tienen el fin de dilucidar los eventos ocurridos en la dictadura. Para ello, se persiguieron tres objetivos: obtener los mejores resultados en la salida de un OCR para los documentos, reconstruir correctamente las transcripciones manuales a partir del proyecto LUISA y utilizar los dos resultados anteriores para poder mejorar mediante técnicas de PLN las salidas obtenidas por un OCR.

Durante todas las etapas del proyectos se tuvo en cuenta la confidencialidad de la información manejada, evitando utilizar medios *online* para cualquier requerimiento presentado con los documentos enteros como puede ser: procesarlos, editarlos, distribuirlos o almacenarlos.

Para el primer objetivo se probaron y compararon tres OCRs distintos (Tesseract, Abbyy Fine Reader y Readiris), siendo Abbyy Fine Reader el OCR que presentó mejores resultados. Si bien, Abbyy Fine Reader presenta algunos errores reconociendo documentos con baja calidad, estos errores son menos frecuentes que su competencia, sumado a esto, reconoce sellos, manchas e imágenes que no transcribe para no generar ruido.

Abbyy Fine Reader superó a los demás OCRs en diez de los once casos evaluados con el puntaje obtenido por la herramienta SequenceMatcher y las transcripciones de LUISA. Comparando manualmente se comprobó la superioridad indicada por el puntaje. Por otro lado, Abbyy Fine Reader presenta una interfaz gráfica y facilidad para el procesamiento de múltiples documentos a través de una *hot folder*. Esto hace que se perciba a Abbyy Fine Reader como el OCR más adecuado para procesar múltiples documentos de forma fácil, rápida y obteniendo los mejores resultados.

Como segundo objetivo se logró reconstruir documentos simulando transcripciones humanas. Esto se hizo implementando un algoritmo el cual toma como entrada las transcripciones de palabras aisladas realizadas por colaboradores de la plataforma LUISA, y rearma los textos completos haciendo uso de las coordenadas de cada palabra dentro del documento original. Se puso especial atención en intentar seleccionar la mejor transcripción posible, ya que cada palabra podía tener más de una transcripción asociada. Se logró sortear dificultades como la identificación y unión de bloques separados erróneamente, eliminación de bloques pertenecientes a sellos comunes e identificación de saltos de línea entre bloques. Con lo anterior mencionado, se llegó tener en formato texto una transcripción fiel de los documentos digitalizados relativamente cercana a lo que sería una transcripción manual.

El tercer objetivo, el cual conlleva la mayor parte del proyecto, se encaró por dos direcciones distintas: una de ellas consiste en corregir errores por medio de modelos de lenguaje, mientras que el otro realiza la corrección a través de la traducción automática estadística.

Por un lado, la corrección por modelo de lenguaje requirió de la creación de un vocabulario para poder detectar los errores ortográficos y brindar candidatos para corregirlos. También se necesitó un modelo de lenguaje que sea lo suficientemente completo como para ofrecer probabilidades a la infinidad de n-gramas. La eficacia de este método está fuertemente basada en el tamaño del vocabulario y la cantidad de datos utilizados para construir el modelo de lenguaje.

Por otro lado, la corrección por medio de traducción automática estadística requirió la creación de un corpus paralelo y el entrenamiento de un traductor. La creación del corpus paralelo tuvo la dificultad de alinear elementos generados de distinta forma. La eficacia de este método depende en gran medida de la cantidad de datos suministrada, en este caso los datos alineados utilizados fueron escasos en comparación con lo indicado en la documentación.

Si bien, con ambos métodos se logró una corrección de los documentos, las comparaciones mostraron que el método que logró mejores resultados fue el de modelo de n-gramas, aunque esta comparación no es del todo justa dado que el sistema de traducción automática está subentrenado. El modelo de n-gramas mostró un buen desempeño, mientras que el sistema de traducción automática además de un correcto desempeño posee un gran potencial si se entrena con suficientes datos.

Capítulo 9

Trabajos a futuro

Algunos trabajos a futuro que se pueden realizar con el objetivo de realizar mejoras en los resultados obtenidos pueden ser:

Reconstruir LUISA como una matriz

En la reconstrucción actual no se logran conservar los márgenes y el espaciado que presentan los textos, lo que genera que formatos como tablas pierdan su alineación. Utilizando las coordenadas de los bloques y representando los documentos como una matriz, se podría llegar a mantener la estructura de los textos. Lograr esto facilitaría tareas de alineación entre textos generados por OCRs y las reconstrucciones, ya que estos serían más similares. Una alineación más precisa aporta gran valor a enfoques que dependen de datos alineados como lo es la traducción automática.

Generar versión de LUISA uniendo palabras separadas por salto de línea sin guion

Para el proyecto se generaron dos versiones de LUISA, una sin unir palabras separadas por un salto de línea y otra donde se unen las palabras separadas por un guion y salto de línea. Pero en una de las estrategias de corrección testeadas se implementó la capacidad de reconocer y unir palabras divididas por un salto de línea sin necesidad de que haya un guion en medio (esta implementación busca resolver el caso en que el guion existe en el documento pero no fue reconocido por el OCR). Dada estas diferencias entre la estrategia de corrección y las reconstrucciones de LUISA, para que la comparación entre ambos conjuntos sea más justa se debería tener una reconstrucción de LUISA donde se unan palabras con salto de línea sin guion de por medio al igual que en la estrategia de corrección.

Mejorar vocabulario más nombres, apellidos y calles

Para la solución basada en n-gramas, cuanto más completo sea el vocabulario, menos detecciones de errores equivocadas se producirán. Los nombres poco comunes y calles en algunos casos son reconocidos como errores y se intentan corregir modificando la palabra original correcta. Por otro lado, cuanto más palabras tenga el vocabulario más candidatos a corregir un error habrá, y por lo tanto el universo de palabras como posible corrección crece.

Detectar errores semánticamente y no solo por sintaxis (estar en el vocabulario)

Para la corrección basada en modelo de n-gramas, se utilizó un vocabulario en la detección de errores. Este enfoque detecta errores ortográficos pero no errores semánticos, una posible solución para esto es utilizar contexto y probabilidades para la detección de errores, al igual que se hace en la corrección.

Implementar modelo de Error

Se podría implementar un modelo de errores el cual dada una frase o palabra escritas de forma correcta devuelva la misma frase o palabra conteniendo algún tipo de error. Esto se ha utilizado en varios trabajos similares con dos objetivos principales: Generación de ejemplos de entrenamiento artificiales y evaluación de los correctores.

Entrenar SMT con mayor cantidad de datos

La solución basada en sistema de traducción automática estadística o SMT, obtuvo resultados alentadores a pesar de haber sido entrenado con tan solo 6.000 y 15.000 ejemplos aproximadamente en sus dos enfoques (líneas y oraciones). La documentación oficial de Moses brinda como ejemplo un conjunto de 140.000 ejemplos y aclara que el mismo es muy pequeño. Un trabajo a futuro es el de lograr conseguir una mayor cantidad de datos con los que entrenar más apropiadamente al sistema de traducción.

Exploración de otras soluciones

La variedad de enfoques con los que se puede abordar este tipo de tareas es muy grande. Hablando tanto de las estrategias principales a seguir (como pueden ser corregir por modelos de lenguajes o por traducción automática estadística), como de módulos adicionales más pequeños que resuelven problemas más específicos (algoritmo de alineación, algoritmo de generación de candidatos, etc.). Es un terreno en el que se puede experimentar mucho y en el cual siempre están surgiendo ideas a investigar que suelen resultar en metodologías innovadoras. Como gran área a destacar que no fue explorada en este proyecto pero encontramos sumamente interesante tenemos a los modelos basados en redes neuronales.

Bibliografía

[1] Chirag Patel, Atul Patel y Dharmendra Patel. "Optical Character Recognition by Open source OCR Tool Tesseract: A Case Study". En: *International Journal of Computer Applications* 55 (oct. de 2012), págs. 50-56. DOI: 10.5120/8794-2784.

[2] Dr. Yusuf Perwej y col. "An Overview and Applications of Optical Character Recognition". En: *International Journal of Advance Research In Science And Engineering (IJARSE)* Vol. 3 (jun. de 2014), Pages 261-274.

[3] Noman Islam, Zeeshan Islam y Nazia Noor. "A Survey on Optical Character Recognition System". En: *ITB Journal of Information and Communication Technology* (dic. de 2016).

[4] G. Vamvakas y col. "A Complete Optical Character Recognition Methodology for Historical Documents". En: *Document Analysis Systems, IAPR International Workshop on* 0 (sep. de 2008), págs. 525-532. DOI: 10.1109/DAS.2008.73.

[5] Daniel Jurafsky James H. Martin. "Speech and Language Processing". En: (2019), págs. 1-6.

[6] Greg Corrado y Jeffrey Dean Tomas Mikolov Kai Chen. "Efficient Estimation of Word Representations inVector Space". En: (ene. de 2013), págs. 1-12.

[7] Kai Chen Greg Corrado y Jeffrey Dean Tomas Mikolov Ilya Sutskever. "Distributed Representations of Words and Phrasesand their Compositionality". En: (oct. de 2013), págs. 1-9.

[8] Philipp Koehn. *Statistical Machine Translation*. Ene. de 2009. DOI: 10.1017/CBO9780511815829.

[9] Peter Brown y col. "The Mathematics of Statistical Machine Translation: Parameter Estimation". En: *Computational Linguistics* 19 (ene. de 1993), págs. 263-311.

[10] "Difflib SeuenceMatcher". En: (ago. de 2019).

[11] Kishore Papineni y col. "BLEU: a Method for Automatic Evaluation of Machine Translation". En: (oct. de 2002). DOI: 10.3115/1073083.1073135.

[12] Ahmad Abdulkader y Mathew Casey. "Low Cost Correction of OCR Errors Using Learning in a Multi-Engine Environment". En: ene. de 2009, págs. 576-580. DOI: 10.1109/ICDAR.2009.242.

[13] Xiang Tong y David Evans. "A statistical approach to automatic OCR error correction in context". En: *Proceedings of the Fourth Workshop on Very Large Corpora* (ene. de 1996).

[14] Martin Volk, Lenz Furrer y Rico Sennrich. "Strategies for Reducing and Correcting OCR Errors". En: ene. de 2011, págs. 3-22. DOI: 10.1007/978-3-642-20227-8_1.

[15] Okan Kolak, William Byrne y Ps Resnik. "A generative probabilistic OCR model for NLP applications". En: (mar. de 2004). DOI: 10.3115/1073445.1073463.

[16] Haithem Afli y col. "Using SMT for OCR Error Correction of Historical Texts". En: *Proceedings of LREC-2016* (mayo de 2016).

[17] *Tesseract 4*. https://github.com/tesseract-ocr/tesseract. Abr. de 2019.

[18] *Vietocr*. http://vietocr.sourceforge.net/usage.html. Sep. de 2020.

[19] *OCRFeeder*. https://wiki.gnome.org/Apps/OCRFeeder. Sep. de 2020.

[20] *Tesseract 4 Tesstrain*. https://github.com/tesseract-ocr/tesstrain. Mayo de 2019.

[21] *Leptonica*. https://github.com/DanBloomberg/leptonica. Mayo de 2019.

[22] *ImageMagick*. https://imagemagick.org/index.php. Mayo de 2019.

[23] *ABBYY FineReader 15*. https://www.abbyy.com/es/finereader/. Mayo de 2019.

[24] *READIRIS 17*. https://www.irislink.com/ES/c1729/Readiris-17--la-solucion-PDF-y-OCR-para-Windows-.aspx. Mayo de 2019.

[25] Free Software Foundation. *FreeLing*. http://nlp.lsi.upc.edu/freeling/node/1. 2007.

[26] RLAES. *RLAES*. https://github.com/sbosio/rla-es. 2008.

[27] Rodrigo Lastra. "Full weights dump for ELMo biLM for Spanish language". En: (oct. de 2019).

[28] Lluís Padró y Evgeny Stanilovsky. "FreeLing 3.0: Towards Wider Multilinguality". En: *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*. ELRA. Istanbul, Turkey, mayo de 2012.

[29] Santiago Bosio. "Recursos Lingüísticos Abiertos del Español". En: (oct. de 2019).

[30] Jean-Baptiste Michel y col. "Quantitative Analysis of Culture Using Millions of Digitized Books". En: *Science (New York, N.Y.)* 331 (ene. de 2011), págs. 176-82. DOI: 10.1126/science.1199644.

[31] Cristian Cardellino. *Spanish Billion Words Corpus and Embeddings*. Ago. de 2019. URL: https://crscardellino.github.io/SBWCE/.

[32] Wolf Garbe. *SymSpell*. https://github.com/wolfgarbe/symspell. 2019.

[33] Martin Trenkmann. *PhraseFinder – Search millions of books for language use*. URL: https://phrasefinder.io (visitado 22-06-2020).

[34] *Cristian Cardellino*. https://crscardellino.github.io/.

[35] Kenneth Heafield. "KenLM: Faster and Smaller Language Model Queries". En: (jul. de 2011), págs. 1-3.

[36] Laura Alonso y Irene Castellon Gloria Vàzquez Ana Fernandez. *SENSEM Corpus*. http://grial.uab.es/. 2016.

[37] Universitat d'Alacant y Euskal Herriko Unibertsitatea Universitat de Barcelona Universitat Politècnica de Catalunya. *AnCora Corpus*. http://clic.ub.edu/corpus/en. 2008.

[38] Universitat Pompeu Fabra. *Tibidabo Treebank and IULA Spanish LSP Treebank Train and Test Partitions*. https://github.com/wolfgarbe/symspell. 2010.

[39] Jörg Tiedemann. *OPUS Project Corpora*. http://opus.nlpl.eu/index.php. 2015.

[40] Philipp Koehn. *European Parliament Proceedings Parallel Corpus*. http://statmt.org/europarl/. 2011.

[41] Comunidad. *Wikipedia*. https://es.wikipedia.org/wiki/Wikipedia:Portada. 2015.

[42] Comunidad. *Wikisource*. https://es.wikisource.org/wiki/Portada. 2015.

[43] Comunidad. *Wikilibros*. https://es.wikibooks.org/wiki/Portada. 2015.

[44] MediaWiki. *Wikipedia Extractor*. http://medialab.di.unipi.it/wiki/Wikipedia_Extractor. 2015.

[45] Mohit Iyyer Matt Gardner Christopher Clark Kenton Lee Luke Zettlemoyer Matthew E. Peters Mark Neumann. "Deep contextualized word representations". En: (mar. de 2018), págs. 1-3.

[46] Zheng Yuan y Mariano Felice. "Constrained Grammatical Error Correction using Statistical Machine Translation". En: ago. de 2013, págs. 52-61.

[47] Steven Bird, Ewan Klein y Edward Loper. *Natural Language Processing with Python*. Ene. de 2009. ISBN: 978-0-596-51649-9.

[48] *Bleualign*. https://github.com/rsennrich/Bleualign. 2013.

[49] William Gale y Kenneth Church. "A Program for Aligning Sentences in Bilingual Corpora." En: *Computational Linguistics* 19 (ene. de 1993), págs. 75-102.

[50] Philipp Koehn y col. "Moses: Open Source Toolkit for Statistical Machine Translation." En: jun. de 2007.

[51] Franz Josef y Hermann Ney. "A Systematic Comparison of Various Statistical Alignment Models". En: *Computational Linguistics* 29 (mar. de 2003). DOI: 10.1162/089120103321337421.

[52] Philipp Koehn, Franz Och y Daniel Marcu. "Statistical Phrase-Based Translation." En: ene. de 2003. DOI: 10.3115/1073445.1073462.

[53] Haithem Afli, Loïc Barrault y Holger Schwenk. "OCR Error Correction Using Statistical Machine Translation". En: abr. de 2015.

[54] Google Cloud. *Evaluating models*. https://cloud.google.com/translate/automl/docs/evaluate. 2020.

Capítulo 10

Anexo

10.1. Esquema del proyecto

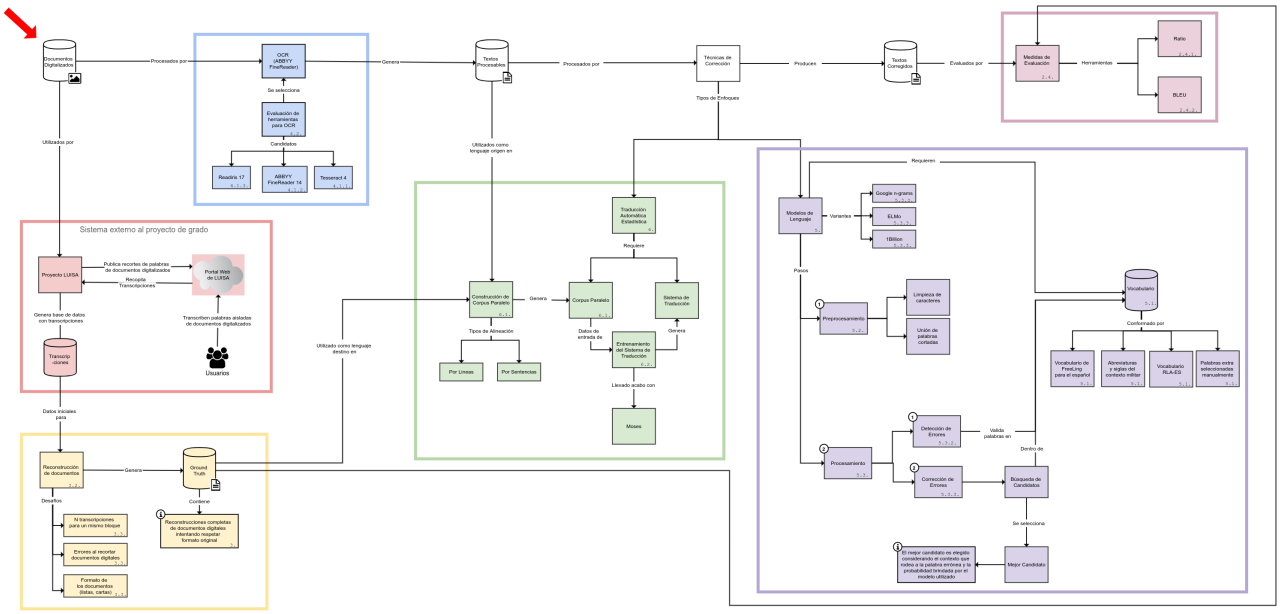


Imagen 10.1: Esquema completo del proyecto

10.2. Pseudocódigo de las reconstrucciones

```
# Distancias en píxeles a considerar para juntar palabras
distancias = [3,4,8,12,16,20,24]
bloques_a_unir = []

# Para cada distancia dist seleccionada
for dist en distancias do
    # Retorna conjuntos de bloques con separación máxima en píxeles de hasta distancia dist para toda la tanda
    bloques_en_dist = obtener_conjunto_bloques_en_distancia(dist, bloques_tanda)

    # Clasifica los conjuntos de bloques en cuatro categorías
    corrects, u_corrects, u_wrong, wrong = clasificar_conjunto_de_bloques(bloques_en_dist)
    bloques_seleccionados = unir(corrects, u_corrects)
    bloques_a_unir[dist] = bloques_seleccionados
end for
# Se consolidan los conjuntos de bloques de distintas distancias en una sola lista, si un conjunto de bloques tiene bloques en común con otro conjunto, entonces se descarta el conjunto que se haya obtenido con una distancia menor
lista_bloques_a_unir = consolidar_en_orden(bloques_a_unir)
hojas = obtener_hojas_tanda()

# Para cada hoja de la tanda
for hoja en hojas do
    transcripcion_hoja = ""
    # Cantidad de bloques a saltear
    evitar_n_bloques = 0

    # La función obtener_bloques_de_hoja evita devolver los bloques que representan sellos en los documentos y los que tienen transcirpciones vacías
    bloques_hojas = obtener_bloques_de_hoja(hoja)
    for bloque_indice en largo(bloques_hoja) do
        if evitar_n_bloques > 0  then
            evitar_n_bloques-= 1
        else
            bloque = bloques_hoja[bloque_indice]
            # Se obtiene la transcripción más frecuente para el bloque
            transcripcion_bloque = obtener_mejor_transcripcion(bloque)
            bloque_siguiente = bloques_hoja[bloque_indice+1]
            transcripcion_bloque_siguiente = obtener_mejor_transcripcion(bloque_siguiente)

            # Si el bloque pertenece a una palabra a unir
            if bloque_en_algun_conjunto_a_unir(bloque, lista_bloques_a_unir)  then
                transcripcion_conjunto = obtener_transcripcion_conjunto_bloque(bloque,lista_bloques_a_unir)
                transcripcion_hoja += " "+ transcripcion_conjunto
                evitar_n_bloques = largo(obtener_conjunto_bloques(bloque)) -1
            else
```

```
                # Si se requiere unir las palabras separadas por un guion y salto de línea
                # Si el bloque termina en guion '-'
                # Si se forma una palabra correcta uniendo el bloque act. sin guion y el bloque sig.
                if (bandera_unir_con_guion) && (bloque_termina_en_guion(bloque)
&& (es_correcto(unir(remover_guion(transcripcion_bloque),transcripcion_bloque_siguiente)) then
                    transcripcion_bloques_unidos = unir(remover_guion(transcripcion_bloque),
transcripcion_bloque_siguiente)
                    transcripcion_hoja += " "+ transcripcion_bloque_unido
                    evitar_n_bloques = 1
                else
                    # El bloque no es una palabra dividida ni está separada por un guion y salto de
línea, por lo tanto, unicamente se agrega su mejor transcripción
                    transcripcion_hoja += " "+ transcripcion_bloque
                end if
            end if
            # Agrega salto de línea si el bloque actual esta totalmente por encima del bloque si-
guiente
            trascripcion_hoja += agregar_salto_linea_si_es_necesario(bloque,bloque_siguiente)
        end if
    end for
    guardar_transcripcion_hoja(transcripcion_hoja)
end for
```

10.3. Pseudocódigo Bleualign

```
for UA_s, UA_t en S * T do
    parejas_y_puntajes += calcular_puntaje_bleu(n-gramas_s_hasta_2, n-gramas_t_hasta_2)
end for
# Se podan las UAs candidatas, conservando únicamente las 3 de mejor puntaje
parejas_y_puntajes = poda_top_3(parejas_y_puntajes)

# Se busca el camino más largo que mantenga el orden monotónico. El resultado es un conjunto
de pares de alineaciones
puntos_de_anclaje = camino_maximal(parejas_y_puntajes)

# UAs restantes sin alinear
huecos_s = [UA_s en S y no en puntos_de_anclaje]
huecos_t = [UA_t en T y no en puntos_de_anclaje]

# Recalculo puntajes Bleu si corresponde
if bleu1to1 en gapfillheuristics then
    for hueco_s, hueco_t en huecos_s * huecos_t do
        # Posibles alineaciones eliminadas en la poda
        parejas_y_puntajes_huecos += calcular_puntaje_bleu(n-gramas_hueco_s_hasta_2,
                                                           n-gramas_hueco_t_hasta_2)
    end for
    huecos_alineados.add(pareja en parejas_y_puntajes_huecos y maximiza_bleu(pareja))
end if

# Galechurch si corresponde
if  galechurch en gapfillheuristics &
   tamaño_mayor_dos(huecos_s_restantes, huecos_t_restantes) then
    huecos_alineados.add(galechurch(hueco_s_restantes, huecos_t_restantes))
end if

resultado = puntos_de_anclaje + huecos_alineados
```

10.4. Configuraciones modelo de lenguaje

Tabla 10.1: Valores de las variables de cada configuración de la prueba en la Subsección 5.3.1.

Configuración	vocabulary	c_u_c	c_u_c_f_l	c_d	p_s_w_b_n	c_s_w	l_m
config1	vocabulary1.5	FALSE	FALSE	forward	split	any_line	google
config2	vocabulary1.5	FALSE	FALSE	forward	split	same_line	google
config3	vocabulary1.5	FALSE	FALSE	forward	split	not_process	google
config4	vocabulary1.5	FALSE	FALSE	forward	split	any_line	1_billion
config5	vocabulary1.5	FALSE	FALSE	forward	split	same_line	1_billion
config6	vocabulary1.5	FALSE	FALSE	forward	split	not_process	1_billion
config7	vocabulary1.5	FALSE	FALSE	previous	split	any_line	google
config8	vocabulary1.5	FALSE	FALSE	previous	split	same_line	google
config9	vocabulary1.5	FALSE	FALSE	previous	split	not_process	google
config10	vocabulary1.5	FALSE	FALSE	previous	split	any_line	1_billion
config11	vocabulary1.5	FALSE	FALSE	previous	split	same_line	1_billion
config12	vocabulary1.5	FALSE	FALSE	previous	split	not_process	1_billion
config13	vocabulary1.5	FALSE	FALSE	middle	split	any_line	google
config14	vocabulary1.5	FALSE	FALSE	middle	split	same_line	google
config15	vocabulary1.5	FALSE	FALSE	middle	split	not_process	google
config16	vocabulary1.5	FALSE	FALSE	middle	split	any_line	1_billion
config17	vocabulary1.5	FALSE	FALSE	middle	split	same_line	1_billion
config18	vocabulary1.5	FALSE	FALSE	middle	split	not_process	1_billion
config19	vocabulary1.5	FALSE	FALSE	all	split	any_line	google
config20	vocabulary1.5	FALSE	FALSE	all	split	same_line	google
config21	vocabulary1.5	FALSE	FALSE	all	split	not_process	google
config22	vocabulary1.5	FALSE	FALSE	all	split	any_line	1_billion
config23	vocabulary1.5	FALSE	FALSE	all	split	same_line	1_billion
config24	vocabulary1.5	FALSE	FALSE	all	split	not_process	1_billion
config25	vocabulary1.5	FALSE	FALSE	forward	join	any_line	google
config26	vocabulary1.5	FALSE	FALSE	forward	join	same_line	google
config27	vocabulary1.5	FALSE	FALSE	forward	join	not_process	google
config28	vocabulary1.5	FALSE	FALSE	forward	join	any_line	1_billion
config29	vocabulary1.5	FALSE	FALSE	forward	join	same_line	1_billion
config30	vocabulary1.5	FALSE	FALSE	forward	join	not_process	1_billion
config31	vocabulary1.5	FALSE	FALSE	previous	join	any_line	google
config32	vocabulary1.5	FALSE	FALSE	previous	join	same_line	google
config33	vocabulary1.5	FALSE	FALSE	previous	join	not_process	google
config34	vocabulary1.5	FALSE	FALSE	previous	join	any_line	1_billion
config35	vocabulary1.5	FALSE	FALSE	previous	join	same_line	1_billion
config36	vocabulary1.5	FALSE	FALSE	previous	join	not_process	1_billion
config37	vocabulary1.5	FALSE	FALSE	middle	join	any_line	google
config38	vocabulary1.5	FALSE	FALSE	middle	join	same_line	google
config39	vocabulary1.5	FALSE	FALSE	middle	join	not_process	google
config40	vocabulary1.5	FALSE	FALSE	middle	join	any_line	1_billion
config41	vocabulary1.5	FALSE	FALSE	middle	join	same_line	1_billion

config42	vocabulary1.5	FALSE	FALSE	middle	join	not_process	1_billion
config43	vocabulary1.5	FALSE	FALSE	all	join	any_line	google
config44	vocabulary1.5	FALSE	FALSE	all	join	same_line	google
config45	vocabulary1.5	FALSE	FALSE	all	join	not_process	google
config46	vocabulary1.5	FALSE	FALSE	all	join	any_line	1_billion
config47	vocabulary1.5	FALSE	FALSE	all	join	same_line	1_billion
config48	vocabulary1.5	FALSE	FALSE	all	join	not_process	1_billion
config49	vocabulary1.5	FALSE	FALSE	forward	not_process	any_line	google
config50	vocabulary1.5	FALSE	FALSE	forward	not_process	same_line	google
config51	vocabulary1.5	FALSE	FALSE	forward	not_process	not_process	google
config52	vocabulary1.5	FALSE	FALSE	forward	not_process	any_line	1_billion
config53	vocabulary1.5	FALSE	FALSE	forward	not_process	same_line	1_billion
config54	vocabulary1.5	FALSE	FALSE	forward	not_process	not_process	1_billion
config55	vocabulary1.5	FALSE	FALSE	previous	not_process	any_line	google
config56	vocabulary1.5	FALSE	FALSE	previous	not_process	same_line	google
config57	vocabulary1.5	FALSE	FALSE	previous	not_process	not_process	google
config58	vocabulary1.5	FALSE	FALSE	previous	not_process	any_line	1_billion
config59	vocabulary1.5	FALSE	FALSE	previous	not_process	same_line	1_billion
config60	vocabulary1.5	FALSE	FALSE	previous	not_process	not_process	1_billion
config61	vocabulary1.5	FALSE	FALSE	middle	not_process	any_line	google
config62	vocabulary1.5	FALSE	FALSE	middle	not_process	same_line	google
config63	vocabulary1.5	FALSE	FALSE	middle	not_process	not_process	google
config64	vocabulary1.5	FALSE	FALSE	middle	not_process	any_line	1_billion
config65	vocabulary1.5	FALSE	FALSE	middle	not_process	same_line	1_billion
config66	vocabulary1.5	FALSE	FALSE	middle	not_process	not_process	1_billion
config67	vocabulary1.5	FALSE	FALSE	all	not_process	any_line	google
config68	vocabulary1.5	FALSE	FALSE	all	not_process	same_line	google
config69	vocabulary1.5	FALSE	FALSE	all	not_process	not_process	google
config70	vocabulary1.5	FALSE	FALSE	all	not_process	any_line	1_billion
config71	vocabulary1.5	FALSE	FALSE	all	not_process	same_line	1_billion
config72	vocabulary1.5	FALSE	FALSE	all	not_process	not_process	1_billion
config73	vocabulary1.5	FALSE	TRUE	forward	split	any_line	google
config74	vocabulary1.5	FALSE	TRUE	forward	split	same_line	google
config75	vocabulary1.5	FALSE	TRUE	forward	split	not_process	google
config76	vocabulary1.5	FALSE	TRUE	forward	split	any_line	1_billion
config77	vocabulary1.5	FALSE	TRUE	forward	split	same_line	1_billion
config78	vocabulary1.5	FALSE	TRUE	forward	split	not_process	1_billion
config79	vocabulary1.5	FALSE	TRUE	previous	split	any_line	google
config80	vocabulary1.5	FALSE	TRUE	previous	split	same_line	google
config81	vocabulary1.5	FALSE	TRUE	previous	split	not_process	google
config82	vocabulary1.5	FALSE	TRUE	previous	split	any_line	1_billion
config83	vocabulary1.5	FALSE	TRUE	previous	split	same_line	1_billion
config84	vocabulary1.5	FALSE	TRUE	previous	split	not_process	1_billion
config85	vocabulary1.5	FALSE	TRUE	middle	split	any_line	google
config86	vocabulary1.5	FALSE	TRUE	middle	split	same_line	google
config87	vocabulary1.5	FALSE	TRUE	middle	split	not_process	google

config88	vocabulary1.5	FALSE	TRUE	middle	split	any_line	1_billion
config89	vocabulary1.5	FALSE	TRUE	middle	split	same_line	1_billion
config90	vocabulary1.5	FALSE	TRUE	middle	split	not_process	1_billion
config91	vocabulary1.5	FALSE	TRUE	all	split	any_line	google
config92	vocabulary1.5	FALSE	TRUE	all	split	same_line	google
config93	vocabulary1.5	FALSE	TRUE	all	split	not_process	google
config94	vocabulary1.5	FALSE	TRUE	all	split	any_line	1_billion
config95	vocabulary1.5	FALSE	TRUE	all	split	same_line	1_billion
config96	vocabulary1.5	FALSE	TRUE	all	split	not_process	1_billion
config97	vocabulary1.5	FALSE	TRUE	forward	join	any_line	google
config98	vocabulary1.5	FALSE	TRUE	forward	join	same_line	google
config99	vocabulary1.5	FALSE	TRUE	forward	join	not_process	google
config100	vocabulary1.5	FALSE	TRUE	forward	join	any_line	1_billion
config101	vocabulary1.5	FALSE	TRUE	forward	join	same_line	1_billion
config102	vocabulary1.5	FALSE	TRUE	forward	join	not_process	1_billion
config103	vocabulary1.5	FALSE	TRUE	previous	join	any_line	google
config104	vocabulary1.5	FALSE	TRUE	previous	join	same_line	google
config105	vocabulary1.5	FALSE	TRUE	previous	join	not_process	google
config106	vocabulary1.5	FALSE	TRUE	previous	join	any_line	1_billion
config107	vocabulary1.5	FALSE	TRUE	previous	join	same_line	1_billion
config108	vocabulary1.5	FALSE	TRUE	previous	join	not_process	1_billion
config109	vocabulary1.5	FALSE	TRUE	middle	join	any_line	google
config110	vocabulary1.5	FALSE	TRUE	middle	join	same_line	google
config111	vocabulary1.5	FALSE	TRUE	middle	join	not_process	google
config112	vocabulary1.5	FALSE	TRUE	middle	join	any_line	1_billion
config113	vocabulary1.5	FALSE	TRUE	middle	join	same_line	1_billion
config114	vocabulary1.5	FALSE	TRUE	middle	join	not_process	1_billion
config115	vocabulary1.5	FALSE	TRUE	all	join	any_line	google
config116	vocabulary1.5	FALSE	TRUE	all	join	same_line	google
config117	vocabulary1.5	FALSE	TRUE	all	join	not_process	google
config118	vocabulary1.5	FALSE	TRUE	all	join	any_line	1_billion
config119	vocabulary1.5	FALSE	TRUE	all	join	same_line	1_billion
config120	vocabulary1.5	FALSE	TRUE	all	join	not_process	1_billion
config121	vocabulary1.5	FALSE	TRUE	forward	not_process	any_line	google
config122	vocabulary1.5	FALSE	TRUE	forward	not_process	same_line	google
config123	vocabulary1.5	FALSE	TRUE	forward	not_process	not_process	google
config124	vocabulary1.5	FALSE	TRUE	forward	not_process	any_line	1_billion
config125	vocabulary1.5	FALSE	TRUE	forward	not_process	same_line	1_billion
config126	vocabulary1.5	FALSE	TRUE	forward	not_process	not_process	1_billion
config127	vocabulary1.5	FALSE	TRUE	previous	not_process	any_line	google
config128	vocabulary1.5	FALSE	TRUE	previous	not_process	same_line	google
config129	vocabulary1.5	FALSE	TRUE	previous	not_process	not_process	google
config130	vocabulary1.5	FALSE	TRUE	previous	not_process	any_line	1_billion
config131	vocabulary1.5	FALSE	TRUE	previous	not_process	same_line	1_billion
config132	vocabulary1.5	FALSE	TRUE	previous	not_process	not_process	1_billion
config133	vocabulary1.5	FALSE	TRUE	middle	not_process	any_line	google

config134	vocabulary1.5	FALSE	TRUE	middle	not_process	same_line	google
config135	vocabulary1.5	FALSE	TRUE	middle	not_process	not_process	google
config136	vocabulary1.5	FALSE	TRUE	middle	not_process	any_line	1_billion
config137	vocabulary1.5	FALSE	TRUE	middle	not_process	same_line	1_billion
config138	vocabulary1.5	FALSE	TRUE	middle	not_process	not_process	1_billion
config139	vocabulary1.5	FALSE	TRUE	all	not_process	any_line	google
config140	vocabulary1.5	FALSE	TRUE	all	not_process	same_line	google
config141	vocabulary1.5	FALSE	TRUE	all	not_process	not_process	google
config142	vocabulary1.5	FALSE	TRUE	all	not_process	any_line	1_billion
config143	vocabulary1.5	FALSE	TRUE	all	not_process	same_line	1_billion
config144	vocabulary1.5	FALSE	TRUE	all	not_process	not_process	1_billion
config145	vocabulary1.5	TRUE	FALSE	forward	split	any_line	google
config146	vocabulary1.5	TRUE	FALSE	forward	split	same_line	google
config147	vocabulary1.5	TRUE	FALSE	forward	split	not_process	google
config148	vocabulary1.5	TRUE	FALSE	forward	split	any_line	1_billion
config149	vocabulary1.5	TRUE	FALSE	forward	split	same_line	1_billion
config150	vocabulary1.5	TRUE	FALSE	forward	split	not_process	1_billion
config151	vocabulary1.5	TRUE	FALSE	previous	split	any_line	google
config152	vocabulary1.5	TRUE	FALSE	previous	split	same_line	google
config153	vocabulary1.5	TRUE	FALSE	previous	split	not_process	google
config154	vocabulary1.5	TRUE	FALSE	previous	split	any_line	1_billion
config155	vocabulary1.5	TRUE	FALSE	previous	split	same_line	1_billion
config156	vocabulary1.5	TRUE	FALSE	previous	split	not_process	1_billion
config157	vocabulary1.5	TRUE	FALSE	middle	split	any_line	google
config158	vocabulary1.5	TRUE	FALSE	middle	split	same_line	google
config159	vocabulary1.5	TRUE	FALSE	middle	split	not_process	google
config160	vocabulary1.5	TRUE	FALSE	middle	split	any_line	1_billion
config161	vocabulary1.5	TRUE	FALSE	middle	split	same_line	1_billion
config162	vocabulary1.5	TRUE	FALSE	middle	split	not_process	1_billion
config163	vocabulary1.5	TRUE	FALSE	all	split	any_line	google
config164	vocabulary1.5	TRUE	FALSE	all	split	same_line	google
config165	vocabulary1.5	TRUE	FALSE	all	split	not_process	google
config166	vocabulary1.5	TRUE	FALSE	all	split	any_line	1_billion
config167	vocabulary1.5	TRUE	FALSE	all	split	same_line	1_billion
config168	vocabulary1.5	TRUE	FALSE	all	split	not_process	1_billion
config169	vocabulary1.5	TRUE	FALSE	forward	join	any_line	google
config170	vocabulary1.5	TRUE	FALSE	forward	join	same_line	google
config171	vocabulary1.5	TRUE	FALSE	forward	join	not_process	google
config172	vocabulary1.5	TRUE	FALSE	forward	join	any_line	1_billion
config173	vocabulary1.5	TRUE	FALSE	forward	join	same_line	1_billion
config174	vocabulary1.5	TRUE	FALSE	forward	join	not_process	1_billion
config175	vocabulary1.5	TRUE	FALSE	previous	join	any_line	google
config176	vocabulary1.5	TRUE	FALSE	previous	join	same_line	google
config177	vocabulary1.5	TRUE	FALSE	previous	join	not_process	google
config178	vocabulary1.5	TRUE	FALSE	previous	join	any_line	1_billion
config179	vocabulary1.5	TRUE	FALSE	previous	join	same_line	1_billion

config180	vocabulary1.5	TRUE	FALSE	previous	join	not_process	1_billion
config181	vocabulary1.5	TRUE	FALSE	middle	join	any_line	google
config182	vocabulary1.5	TRUE	FALSE	middle	join	same_line	google
config183	vocabulary1.5	TRUE	FALSE	middle	join	not_process	google
config184	vocabulary1.5	TRUE	FALSE	middle	join	any_line	1_billion
config185	vocabulary1.5	TRUE	FALSE	middle	join	same_line	1_billion
config186	vocabulary1.5	TRUE	FALSE	middle	join	not_process	1_billion
config187	vocabulary1.5	TRUE	FALSE	all	join	any_line	google
config188	vocabulary1.5	TRUE	FALSE	all	join	same_line	google
config189	vocabulary1.5	TRUE	FALSE	all	join	not_process	google
config190	vocabulary1.5	TRUE	FALSE	all	join	any_line	1_billion
config191	vocabulary1.5	TRUE	FALSE	all	join	same_line	1_billion
config192	vocabulary1.5	TRUE	FALSE	all	join	not_process	1_billion
config193	vocabulary1.5	TRUE	FALSE	forward	not_process	any_line	google
config194	vocabulary1.5	TRUE	FALSE	forward	not_process	same_line	google
config195	vocabulary1.5	TRUE	FALSE	forward	not_process	not_process	google
config196	vocabulary1.5	TRUE	FALSE	forward	not_process	any_line	1_billion
config197	vocabulary1.5	TRUE	FALSE	forward	not_process	same_line	1_billion
config198	vocabulary1.5	TRUE	FALSE	forward	not_process	not_process	1_billion
config199	vocabulary1.5	TRUE	FALSE	previous	not_process	any_line	google
config200	vocabulary1.5	TRUE	FALSE	previous	not_process	same_line	google
config201	vocabulary1.5	TRUE	FALSE	previous	not_process	not_process	google
config202	vocabulary1.5	TRUE	FALSE	previous	not_process	any_line	1_billion
config203	vocabulary1.5	TRUE	FALSE	previous	not_process	same_line	1_billion
config204	vocabulary1.5	TRUE	FALSE	previous	not_process	not_process	1_billion
config205	vocabulary1.5	TRUE	FALSE	middle	not_process	any_line	google
config206	vocabulary1.5	TRUE	FALSE	middle	not_process	same_line	google
config207	vocabulary1.5	TRUE	FALSE	middle	not_process	not_process	google
config208	vocabulary1.5	TRUE	FALSE	middle	not_process	any_line	1_billion
config209	vocabulary1.5	TRUE	FALSE	middle	not_process	same_line	1_billion
config210	vocabulary1.5	TRUE	FALSE	middle	not_process	not_process	1_billion
config211	vocabulary1.5	TRUE	FALSE	all	not_process	any_line	google
config212	vocabulary1.5	TRUE	FALSE	all	not_process	same_line	google
config213	vocabulary1.5	TRUE	FALSE	all	not_process	not_process	google
config214	vocabulary1.5	TRUE	FALSE	all	not_process	any_line	1_billion
config215	vocabulary1.5	TRUE	FALSE	all	not_process	same_line	1_billion
config216	vocabulary1.5	TRUE	FALSE	all	not_process	not_process	1_billion
config217	vocabulary1.5	TRUE	TRUE	forward	split	any_line	google
config218	vocabulary1.5	TRUE	TRUE	forward	split	same_line	google
config219	vocabulary1.5	TRUE	TRUE	forward	split	not_process	google
config220	vocabulary1.5	TRUE	TRUE	forward	split	any_line	1_billion
config221	vocabulary1.5	TRUE	TRUE	forward	split	same_line	1_billion
config222	vocabulary1.5	TRUE	TRUE	forward	split	not_process	1_billion
config223	vocabulary1.5	TRUE	TRUE	previous	split	any_line	google
config224	vocabulary1.5	TRUE	TRUE	previous	split	same_line	google
config225	vocabulary1.5	TRUE	TRUE	previous	split	not_process	google

config226	vocabulary1.5	TRUE	TRUE	previous	split	any_line	1_billion
config227	vocabulary1.5	TRUE	TRUE	previous	split	same_line	1_billion
config228	vocabulary1.5	TRUE	TRUE	previous	split	not_process	1_billion
config229	vocabulary1.5	TRUE	TRUE	middle	split	any_line	google
config230	vocabulary1.5	TRUE	TRUE	middle	split	same_line	google
config231	vocabulary1.5	TRUE	TRUE	middle	split	not_process	google
config232	vocabulary1.5	TRUE	TRUE	middle	split	any_line	1_billion
config233	vocabulary1.5	TRUE	TRUE	middle	split	same_line	1_billion
config234	vocabulary1.5	TRUE	TRUE	middle	split	not_process	1_billion
config235	vocabulary1.5	TRUE	TRUE	all	split	any_line	google
config236	vocabulary1.5	TRUE	TRUE	all	split	same_line	google
config237	vocabulary1.5	TRUE	TRUE	all	split	not_process	google
config238	vocabulary1.5	TRUE	TRUE	all	split	any_line	1_billion
config239	vocabulary1.5	TRUE	TRUE	all	split	same_line	1_billion
config240	vocabulary1.5	TRUE	TRUE	all	split	not_process	1_billion
config241	vocabulary1.5	TRUE	TRUE	forward	join	any_line	google
config242	vocabulary1.5	TRUE	TRUE	forward	join	same_line	google
config243	vocabulary1.5	TRUE	TRUE	forward	join	not_process	google
config244	vocabulary1.5	TRUE	TRUE	forward	join	any_line	1_billion
config245	vocabulary1.5	TRUE	TRUE	forward	join	same_line	1_billion
config246	vocabulary1.5	TRUE	TRUE	forward	join	not_process	1_billion
config247	vocabulary1.5	TRUE	TRUE	previous	join	any_line	google
config248	vocabulary1.5	TRUE	TRUE	previous	join	same_line	google
config249	vocabulary1.5	TRUE	TRUE	previous	join	not_process	google
config250	vocabulary1.5	TRUE	TRUE	previous	join	any_line	1_billion
config251	vocabulary1.5	TRUE	TRUE	previous	join	same_line	1_billion
config252	vocabulary1.5	TRUE	TRUE	previous	join	not_process	1_billion
config253	vocabulary1.5	TRUE	TRUE	middle	join	any_line	google
config254	vocabulary1.5	TRUE	TRUE	middle	join	same_line	google
config255	vocabulary1.5	TRUE	TRUE	middle	join	not_process	google
config256	vocabulary1.5	TRUE	TRUE	middle	join	any_line	1_billion
config257	vocabulary1.5	TRUE	TRUE	middle	join	same_line	1_billion
config258	vocabulary1.5	TRUE	TRUE	middle	join	not_process	1_billion
config259	vocabulary1.5	TRUE	TRUE	all	join	any_line	google
config260	vocabulary1.5	TRUE	TRUE	all	join	same_line	google
config261	vocabulary1.5	TRUE	TRUE	all	join	not_process	google
config262	vocabulary1.5	TRUE	TRUE	all	join	any_line	1_billion
config263	vocabulary1.5	TRUE	TRUE	all	join	same_line	1_billion
config264	vocabulary1.5	TRUE	TRUE	all	join	not_process	1_billion
config265	vocabulary1.5	TRUE	TRUE	forward	not_process	any_line	google
config266	vocabulary1.5	TRUE	TRUE	forward	not_process	same_line	google
config267	vocabulary1.5	TRUE	TRUE	forward	not_process	not_process	google
config268	vocabulary1.5	TRUE	TRUE	forward	not_process	any_line	1_billion
config269	vocabulary1.5	TRUE	TRUE	forward	not_process	same_line	1_billion
config270	vocabulary1.5	TRUE	TRUE	forward	not_process	not_process	1_billion
config271	vocabulary1.5	TRUE	TRUE	previous	not_process	any_line	google

config272	vocabulary1.5	TRUE	TRUE	previous	not_process	same_line	google
config273	vocabulary1.5	TRUE	TRUE	previous	not_process	not_process	google
config274	vocabulary1.5	TRUE	TRUE	previous	not_process	any_line	1_billion
config275	vocabulary1.5	TRUE	TRUE	previous	not_process	same_line	1_billion
config276	vocabulary1.5	TRUE	TRUE	previous	not_process	not_process	1_billion
config277	vocabulary1.5	TRUE	TRUE	middle	not_process	any_line	google
config278	vocabulary1.5	TRUE	TRUE	middle	not_process	same_line	google
config279	vocabulary1.5	TRUE	TRUE	middle	not_process	not_process	google
config280	vocabulary1.5	TRUE	TRUE	middle	not_process	any_line	1_billion
config281	vocabulary1.5	TRUE	TRUE	middle	not_process	same_line	1_billion
config282	vocabulary1.5	TRUE	TRUE	middle	not_process	not_process	1_billion
config283	vocabulary1.5	TRUE	TRUE	all	not_process	any_line	google
config284	vocabulary1.5	TRUE	TRUE	all	not_process	same_line	google
config285	vocabulary1.5	TRUE	TRUE	all	not_process	not_process	google
config286	vocabulary1.5	TRUE	TRUE	all	not_process	any_line	1_billion
config287	vocabulary1.5	TRUE	TRUE	all	not_process	same_line	1_billion
config288	vocabulary1.5	TRUE	TRUE	all	not_process	not_process	1_billion
config289	elmo_vocabulary	FALSE	FALSE	forward	split	any_line	google
config290	elmo_vocabulary	FALSE	FALSE	forward	split	same_line	google
config291	elmo_vocabulary	FALSE	FALSE	forward	split	not_process	google
config292	elmo_vocabulary	FALSE	FALSE	forward	split	any_line	1_billion
config293	elmo_vocabulary	FALSE	FALSE	forward	split	same_line	1_billion
config294	elmo_vocabulary	FALSE	FALSE	forward	split	not_process	1_billion
config295	elmo_vocabulary	FALSE	FALSE	previous	split	any_line	google
config296	elmo_vocabulary	FALSE	FALSE	previous	split	same_line	google
config297	elmo_vocabulary	FALSE	FALSE	previous	split	not_process	google
config298	elmo_vocabulary	FALSE	FALSE	previous	split	any_line	1_billion
config299	elmo_vocabulary	FALSE	FALSE	previous	split	same_line	1_billion
config300	elmo_vocabulary	FALSE	FALSE	previous	split	not_process	1_billion
config301	elmo_vocabulary	FALSE	FALSE	middle	split	any_line	google
config302	elmo_vocabulary	FALSE	FALSE	middle	split	same_line	google
config303	elmo_vocabulary	FALSE	FALSE	middle	split	not_process	google
config304	elmo_vocabulary	FALSE	FALSE	middle	split	any_line	1_billion
config305	elmo_vocabulary	FALSE	FALSE	middle	split	same_line	1_billion
config306	elmo_vocabulary	FALSE	FALSE	middle	split	not_process	1_billion
config307	elmo_vocabulary	FALSE	FALSE	all	split	any_line	google
config308	elmo_vocabulary	FALSE	FALSE	all	split	same_line	google
config309	elmo_vocabulary	FALSE	FALSE	all	split	not_process	google
config310	elmo_vocabulary	FALSE	FALSE	all	split	any_line	1_billion
config311	elmo_vocabulary	FALSE	FALSE	all	split	same_line	1_billion
config312	elmo_vocabulary	FALSE	FALSE	all	split	not_process	1_billion
config313	elmo_vocabulary	FALSE	FALSE	forward	join	any_line	google
config314	elmo_vocabulary	FALSE	FALSE	forward	join	same_line	google
config315	elmo_vocabulary	FALSE	FALSE	forward	join	not_process	google
config316	elmo_vocabulary	FALSE	FALSE	forward	join	any_line	1_billion
config317	elmo_vocabulary	FALSE	FALSE	forward	join	same_line	1_billion

config318	elmo_vocabulary	FALSE	FALSE	forward	join	not_process	1_billion
config319	elmo_vocabulary	FALSE	FALSE	previous	join	any_line	google
config320	elmo_vocabulary	FALSE	FALSE	previous	join	same_line	google
config321	elmo_vocabulary	FALSE	FALSE	previous	join	not_process	google
config322	elmo_vocabulary	FALSE	FALSE	previous	join	any_line	1_billion
config323	elmo_vocabulary	FALSE	FALSE	previous	join	same_line	1_billion
config324	elmo_vocabulary	FALSE	FALSE	previous	join	not_process	1_billion
config325	elmo_vocabulary	FALSE	FALSE	middle	join	any_line	google
config326	elmo_vocabulary	FALSE	FALSE	middle	join	same_line	google
config327	elmo_vocabulary	FALSE	FALSE	middle	join	not_process	google
config328	elmo_vocabulary	FALSE	FALSE	middle	join	any_line	1_billion
config329	elmo_vocabulary	FALSE	FALSE	middle	join	same_line	1_billion
config330	elmo_vocabulary	FALSE	FALSE	middle	join	not_process	1_billion
config331	elmo_vocabulary	FALSE	FALSE	all	join	any_line	google
config332	elmo_vocabulary	FALSE	FALSE	all	join	same_line	google
config333	elmo_vocabulary	FALSE	FALSE	all	join	not_process	google
config334	elmo_vocabulary	FALSE	FALSE	all	join	any_line	1_billion
config335	elmo_vocabulary	FALSE	FALSE	all	join	same_line	1_billion
config336	elmo_vocabulary	FALSE	FALSE	all	join	not_process	1_billion
config337	elmo_vocabulary	FALSE	FALSE	forward	not_process	any_line	google
config338	elmo_vocabulary	FALSE	FALSE	forward	not_process	same_line	google
config339	elmo_vocabulary	FALSE	FALSE	forward	not_process	not_process	google
config340	elmo_vocabulary	FALSE	FALSE	forward	not_process	any_line	1_billion
config341	elmo_vocabulary	FALSE	FALSE	forward	not_process	same_line	1_billion
config342	elmo_vocabulary	FALSE	FALSE	forward	not_process	not_process	1_billion
config343	elmo_vocabulary	FALSE	FALSE	previous	not_process	any_line	google
config344	elmo_vocabulary	FALSE	FALSE	previous	not_process	same_line	google
config345	elmo_vocabulary	FALSE	FALSE	previous	not_process	not_process	google
config346	elmo_vocabulary	FALSE	FALSE	previous	not_process	any_line	1_billion
config347	elmo_vocabulary	FALSE	FALSE	previous	not_process	same_line	1_billion
config348	elmo_vocabulary	FALSE	FALSE	previous	not_process	not_process	1_billion
config349	elmo_vocabulary	FALSE	FALSE	middle	not_process	any_line	google
config350	elmo_vocabulary	FALSE	FALSE	middle	not_process	same_line	google
config351	elmo_vocabulary	FALSE	FALSE	middle	not_process	not_process	google
config352	elmo_vocabulary	FALSE	FALSE	middle	not_process	any_line	1_billion
config353	elmo_vocabulary	FALSE	FALSE	middle	not_process	same_line	1_billion
config354	elmo_vocabulary	FALSE	FALSE	middle	not_process	not_process	1_billion
config355	elmo_vocabulary	FALSE	FALSE	all	not_process	any_line	google
config356	elmo_vocabulary	FALSE	FALSE	all	not_process	same_line	google
config357	elmo_vocabulary	FALSE	FALSE	all	not_process	not_process	google
config358	elmo_vocabulary	FALSE	FALSE	all	not_process	any_line	1_billion
config359	elmo_vocabulary	FALSE	FALSE	all	not_process	same_line	1_billion
config360	elmo_vocabulary	FALSE	FALSE	all	not_process	not_process	1_billion
config361	elmo_vocabulary	FALSE	TRUE	forward	split	any_line	google
config362	elmo_vocabulary	FALSE	TRUE	forward	split	same_line	google
config363	elmo_vocabulary	FALSE	TRUE	forward	split	not_process	google

config364	elmo_vocabulary	FALSE	TRUE	forward	split	any_line	1_billion
config365	elmo_vocabulary	FALSE	TRUE	forward	split	same_line	1_billion
config366	elmo_vocabulary	FALSE	TRUE	forward	split	not_process	1_billion
config367	elmo_vocabulary	FALSE	TRUE	previous	split	any_line	google
config368	elmo_vocabulary	FALSE	TRUE	previous	split	same_line	google
config369	elmo_vocabulary	FALSE	TRUE	previous	split	not_process	google
config370	elmo_vocabulary	FALSE	TRUE	previous	split	any_line	1_billion
config371	elmo_vocabulary	FALSE	TRUE	previous	split	same_line	1_billion
config372	elmo_vocabulary	FALSE	TRUE	previous	split	not_process	1_billion
config373	elmo_vocabulary	FALSE	TRUE	middle	split	any_line	google
config374	elmo_vocabulary	FALSE	TRUE	middle	split	same_line	google
config375	elmo_vocabulary	FALSE	TRUE	middle	split	not_process	google
config376	elmo_vocabulary	FALSE	TRUE	middle	split	any_line	1_billion
config377	elmo_vocabulary	FALSE	TRUE	middle	split	same_line	1_billion
config378	elmo_vocabulary	FALSE	TRUE	middle	split	not_process	1_billion
config379	elmo_vocabulary	FALSE	TRUE	all	split	any_line	google
config380	elmo_vocabulary	FALSE	TRUE	all	split	same_line	google
config381	elmo_vocabulary	FALSE	TRUE	all	split	not_process	google
config382	elmo_vocabulary	FALSE	TRUE	all	split	any_line	1_billion
config383	elmo_vocabulary	FALSE	TRUE	all	split	same_line	1_billion
config384	elmo_vocabulary	FALSE	TRUE	all	split	not_process	1_billion
config385	elmo_vocabulary	FALSE	TRUE	forward	join	any_line	google
config386	elmo_vocabulary	FALSE	TRUE	forward	join	same_line	google
config387	elmo_vocabulary	FALSE	TRUE	forward	join	not_process	google
config388	elmo_vocabulary	FALSE	TRUE	forward	join	any_line	1_billion
config389	elmo_vocabulary	FALSE	TRUE	forward	join	same_line	1_billion
config390	elmo_vocabulary	FALSE	TRUE	forward	join	not_process	1_billion
config391	elmo_vocabulary	FALSE	TRUE	previous	join	any_line	google
config392	elmo_vocabulary	FALSE	TRUE	previous	join	same_line	google
config393	elmo_vocabulary	FALSE	TRUE	previous	join	not_process	google
config394	elmo_vocabulary	FALSE	TRUE	previous	join	any_line	1_billion
config395	elmo_vocabulary	FALSE	TRUE	previous	join	same_line	1_billion
config396	elmo_vocabulary	FALSE	TRUE	previous	join	not_process	1_billion
config397	elmo_vocabulary	FALSE	TRUE	middle	join	any_line	google
config398	elmo_vocabulary	FALSE	TRUE	middle	join	same_line	google
config399	elmo_vocabulary	FALSE	TRUE	middle	join	not_process	google
config400	elmo_vocabulary	FALSE	TRUE	middle	join	any_line	1_billion
config401	elmo_vocabulary	FALSE	TRUE	middle	join	same_line	1_billion
config402	elmo_vocabulary	FALSE	TRUE	middle	join	not_process	1_billion
config403	elmo_vocabulary	FALSE	TRUE	all	join	any_line	google
config404	elmo_vocabulary	FALSE	TRUE	all	join	same_line	google
config405	elmo_vocabulary	FALSE	TRUE	all	join	not_process	google
config406	elmo_vocabulary	FALSE	TRUE	all	join	any_line	1_billion
config407	elmo_vocabulary	FALSE	TRUE	all	join	same_line	1_billion
config408	elmo_vocabulary	FALSE	TRUE	all	join	not_process	1_billion
config409	elmo_vocabulary	FALSE	TRUE	forward	not_process	any_line	google

config410	elmo_vocabulary	FALSE	TRUE	forward	not_process	same_line	google
config411	elmo_vocabulary	FALSE	TRUE	forward	not_process	not_process	google
config412	elmo_vocabulary	FALSE	TRUE	forward	not_process	any_line	1_billion
config413	elmo_vocabulary	FALSE	TRUE	forward	not_process	same_line	1_billion
config414	elmo_vocabulary	FALSE	TRUE	forward	not_process	not_process	1_billion
config415	elmo_vocabulary	FALSE	TRUE	previous	not_process	any_line	google
config416	elmo_vocabulary	FALSE	TRUE	previous	not_process	same_line	google
config417	elmo_vocabulary	FALSE	TRUE	previous	not_process	not_process	google
config418	elmo_vocabulary	FALSE	TRUE	previous	not_process	any_line	1_billion
config419	elmo_vocabulary	FALSE	TRUE	previous	not_process	same_line	1_billion
config420	elmo_vocabulary	FALSE	TRUE	previous	not_process	not_process	1_billion
config421	elmo_vocabulary	FALSE	TRUE	middle	not_process	any_line	google
config422	elmo_vocabulary	FALSE	TRUE	middle	not_process	same_line	google
config423	elmo_vocabulary	FALSE	TRUE	middle	not_process	not_process	google
config424	elmo_vocabulary	FALSE	TRUE	middle	not_process	any_line	1_billion
config425	elmo_vocabulary	FALSE	TRUE	middle	not_process	same_line	1_billion
config426	elmo_vocabulary	FALSE	TRUE	middle	not_process	not_process	1_billion
config427	elmo_vocabulary	FALSE	TRUE	all	not_process	any_line	google
config428	elmo_vocabulary	FALSE	TRUE	all	not_process	same_line	google
config429	elmo_vocabulary	FALSE	TRUE	all	not_process	not_process	google
config430	elmo_vocabulary	FALSE	TRUE	all	not_process	any_line	1_billion
config431	elmo_vocabulary	FALSE	TRUE	all	not_process	same_line	1_billion
config432	elmo_vocabulary	FALSE	TRUE	all	not_process	not_process	1_billion
config433	elmo_vocabulary	TRUE	FALSE	forward	split	any_line	google
config434	elmo_vocabulary	TRUE	FALSE	forward	split	same_line	google
config435	elmo_vocabulary	TRUE	FALSE	forward	split	not_process	google
config436	elmo_vocabulary	TRUE	FALSE	forward	split	any_line	1_billion
config437	elmo_vocabulary	TRUE	FALSE	forward	split	same_line	1_billion
config438	elmo_vocabulary	TRUE	FALSE	forward	split	not_process	1_billion
config439	elmo_vocabulary	TRUE	FALSE	previous	split	any_line	google
config440	elmo_vocabulary	TRUE	FALSE	previous	split	same_line	google
config441	elmo_vocabulary	TRUE	FALSE	previous	split	not_process	google
config442	elmo_vocabulary	TRUE	FALSE	previous	split	any_line	1_billion
config443	elmo_vocabulary	TRUE	FALSE	previous	split	same_line	1_billion
config444	elmo_vocabulary	TRUE	FALSE	previous	split	not_process	1_billion
config445	elmo_vocabulary	TRUE	FALSE	middle	split	any_line	google
config446	elmo_vocabulary	TRUE	FALSE	middle	split	same_line	google
config447	elmo_vocabulary	TRUE	FALSE	middle	split	not_process	google
config448	elmo_vocabulary	TRUE	FALSE	middle	split	any_line	1_billion
config449	elmo_vocabulary	TRUE	FALSE	middle	split	same_line	1_billion
config450	elmo_vocabulary	TRUE	FALSE	middle	split	not_process	1_billion
config451	elmo_vocabulary	TRUE	FALSE	all	split	any_line	google
config452	elmo_vocabulary	TRUE	FALSE	all	split	same_line	google
config453	elmo_vocabulary	TRUE	FALSE	all	split	not_process	google
config454	elmo_vocabulary	TRUE	FALSE	all	split	any_line	1_billion
config455	elmo_vocabulary	TRUE	FALSE	all	split	same_line	1_billion

config456	elmo_vocabulary	TRUE	FALSE	all	split	not_process	1_billion
config457	elmo_vocabulary	TRUE	FALSE	forward	join	any_line	google
config458	elmo_vocabulary	TRUE	FALSE	forward	join	same_line	google
config459	elmo_vocabulary	TRUE	FALSE	forward	join	not_process	google
config460	elmo_vocabulary	TRUE	FALSE	forward	join	any_line	1_billion
config461	elmo_vocabulary	TRUE	FALSE	forward	join	same_line	1_billion
config462	elmo_vocabulary	TRUE	FALSE	forward	join	not_process	1_billion
config463	elmo_vocabulary	TRUE	FALSE	previous	join	any_line	google
config464	elmo_vocabulary	TRUE	FALSE	previous	join	same_line	google
config465	elmo_vocabulary	TRUE	FALSE	previous	join	not_process	google
config466	elmo_vocabulary	TRUE	FALSE	previous	join	any_line	1_billion
config467	elmo_vocabulary	TRUE	FALSE	previous	join	same_line	1_billion
config468	elmo_vocabulary	TRUE	FALSE	previous	join	not_process	1_billion
config469	elmo_vocabulary	TRUE	FALSE	middle	join	any_line	google
config470	elmo_vocabulary	TRUE	FALSE	middle	join	same_line	google
config471	elmo_vocabulary	TRUE	FALSE	middle	join	not_process	google
config472	elmo_vocabulary	TRUE	FALSE	middle	join	any_line	1_billion
config473	elmo_vocabulary	TRUE	FALSE	middle	join	same_line	1_billion
config474	elmo_vocabulary	TRUE	FALSE	middle	join	not_process	1_billion
config475	elmo_vocabulary	TRUE	FALSE	all	join	any_line	google
config476	elmo_vocabulary	TRUE	FALSE	all	join	same_line	google
config477	elmo_vocabulary	TRUE	FALSE	all	join	not_process	google
config478	elmo_vocabulary	TRUE	FALSE	all	join	any_line	1_billion
config479	elmo_vocabulary	TRUE	FALSE	all	join	same_line	1_billion
config480	elmo_vocabulary	TRUE	FALSE	all	join	not_process	1_billion
config481	elmo_vocabulary	TRUE	FALSE	forward	not_process	any_line	google
config482	elmo_vocabulary	TRUE	FALSE	forward	not_process	same_line	google
config483	elmo_vocabulary	TRUE	FALSE	forward	not_process	not_process	google
config484	elmo_vocabulary	TRUE	FALSE	forward	not_process	any_line	1_billion
config485	elmo_vocabulary	TRUE	FALSE	forward	not_process	same_line	1_billion
config486	elmo_vocabulary	TRUE	FALSE	forward	not_process	not_process	1_billion
config487	elmo_vocabulary	TRUE	FALSE	previous	not_process	any_line	google
config488	elmo_vocabulary	TRUE	FALSE	previous	not_process	same_line	google
config489	elmo_vocabulary	TRUE	FALSE	previous	not_process	not_process	google
config490	elmo_vocabulary	TRUE	FALSE	previous	not_process	any_line	1_billion
config491	elmo_vocabulary	TRUE	FALSE	previous	not_process	same_line	1_billion
config492	elmo_vocabulary	TRUE	FALSE	previous	not_process	not_process	1_billion
config493	elmo_vocabulary	TRUE	FALSE	middle	not_process	any_line	google
config494	elmo_vocabulary	TRUE	FALSE	middle	not_process	same_line	google
config495	elmo_vocabulary	TRUE	FALSE	middle	not_process	not_process	google
config496	elmo_vocabulary	TRUE	FALSE	middle	not_process	any_line	1_billion
config497	elmo_vocabulary	TRUE	FALSE	middle	not_process	same_line	1_billion
config498	elmo_vocabulary	TRUE	FALSE	middle	not_process	not_process	1_billion
config499	elmo_vocabulary	TRUE	FALSE	all	not_process	any_line	google
config500	elmo_vocabulary	TRUE	FALSE	all	not_process	same_line	google
config501	elmo_vocabulary	TRUE	FALSE	all	not_process	not_process	google

config502	elmo_vocabulary	TRUE	FALSE	all	not_process	any_line	1_billion
config503	elmo_vocabulary	TRUE	FALSE	all	not_process	same_line	1_billion
config504	elmo_vocabulary	TRUE	FALSE	all	not_process	not_process	1_billion
config505	elmo_vocabulary	TRUE	TRUE	forward	split	any_line	google
config506	elmo_vocabulary	TRUE	TRUE	forward	split	same_line	google
config507	elmo_vocabulary	TRUE	TRUE	forward	split	not_process	google
config508	elmo_vocabulary	TRUE	TRUE	forward	split	any_line	1_billion
config509	elmo_vocabulary	TRUE	TRUE	forward	split	same_line	1_billion
config510	elmo_vocabulary	TRUE	TRUE	forward	split	not_process	1_billion
config511	elmo_vocabulary	TRUE	TRUE	previous	split	any_line	google
config512	elmo_vocabulary	TRUE	TRUE	previous	split	same_line	google
config513	elmo_vocabulary	TRUE	TRUE	previous	split	not_process	google
config514	elmo_vocabulary	TRUE	TRUE	previous	split	any_line	1_billion
config515	elmo_vocabulary	TRUE	TRUE	previous	split	same_line	1_billion
config516	elmo_vocabulary	TRUE	TRUE	previous	split	not_process	1_billion
config517	elmo_vocabulary	TRUE	TRUE	middle	split	any_line	google
config518	elmo_vocabulary	TRUE	TRUE	middle	split	same_line	google
config519	elmo_vocabulary	TRUE	TRUE	middle	split	not_process	google
config520	elmo_vocabulary	TRUE	TRUE	middle	split	any_line	1_billion
config521	elmo_vocabulary	TRUE	TRUE	middle	split	same_line	1_billion
config522	elmo_vocabulary	TRUE	TRUE	middle	split	not_process	1_billion
config523	elmo_vocabulary	TRUE	TRUE	all	split	any_line	google
config524	elmo_vocabulary	TRUE	TRUE	all	split	same_line	google
config525	elmo_vocabulary	TRUE	TRUE	all	split	not_process	google
config526	elmo_vocabulary	TRUE	TRUE	all	split	any_line	1_billion
config527	elmo_vocabulary	TRUE	TRUE	all	split	same_line	1_billion
config528	elmo_vocabulary	TRUE	TRUE	all	split	not_process	1_billion
config529	elmo_vocabulary	TRUE	TRUE	forward	join	any_line	google
config530	elmo_vocabulary	TRUE	TRUE	forward	join	same_line	google
config531	elmo_vocabulary	TRUE	TRUE	forward	join	not_process	google
config532	elmo_vocabulary	TRUE	TRUE	forward	join	any_line	1_billion
config533	elmo_vocabulary	TRUE	TRUE	forward	join	same_line	1_billion
config534	elmo_vocabulary	TRUE	TRUE	forward	join	not_process	1_billion
config535	elmo_vocabulary	TRUE	TRUE	previous	join	any_line	google
config536	elmo_vocabulary	TRUE	TRUE	previous	join	same_line	google
config537	elmo_vocabulary	TRUE	TRUE	previous	join	not_process	google
config538	elmo_vocabulary	TRUE	TRUE	previous	join	any_line	1_billion
config539	elmo_vocabulary	TRUE	TRUE	previous	join	same_line	1_billion
config540	elmo_vocabulary	TRUE	TRUE	previous	join	not_process	1_billion
config541	elmo_vocabulary	TRUE	TRUE	middle	join	any_line	google
config542	elmo_vocabulary	TRUE	TRUE	middle	join	same_line	google
config543	elmo_vocabulary	TRUE	TRUE	middle	join	not_process	google
config544	elmo_vocabulary	TRUE	TRUE	middle	join	any_line	1_billion
config545	elmo_vocabulary	TRUE	TRUE	middle	join	same_line	1_billion
config546	elmo_vocabulary	TRUE	TRUE	middle	join	not_process	1_billion
config547	elmo_vocabulary	TRUE	TRUE	all	join	any_line	google

config548	elmo_vocabulary	TRUE	TRUE	all	join	same_line	google
config549	elmo_vocabulary	TRUE	TRUE	all	join	not_process	google
config550	elmo_vocabulary	TRUE	TRUE	all	join	any_line	1_billion
config551	elmo_vocabulary	TRUE	TRUE	all	join	same_line	1_billion
config552	elmo_vocabulary	TRUE	TRUE	all	join	not_process	1_billion
config553	elmo_vocabulary	TRUE	TRUE	forward	not_process	any_line	google
config554	elmo_vocabulary	TRUE	TRUE	forward	not_process	same_line	google
config555	elmo_vocabulary	TRUE	TRUE	forward	not_process	not_process	google
config556	elmo_vocabulary	TRUE	TRUE	forward	not_process	any_line	1_billion
config557	elmo_vocabulary	TRUE	TRUE	forward	not_process	same_line	1_billion
config558	elmo_vocabulary	TRUE	TRUE	forward	not_process	not_process	1_billion
config559	elmo_vocabulary	TRUE	TRUE	previous	not_process	any_line	google
config560	elmo_vocabulary	TRUE	TRUE	previous	not_process	same_line	google
config561	elmo_vocabulary	TRUE	TRUE	previous	not_process	not_process	google
config562	elmo_vocabulary	TRUE	TRUE	previous	not_process	any_line	1_billion
config563	elmo_vocabulary	TRUE	TRUE	previous	not_process	same_line	1_billion
config564	elmo_vocabulary	TRUE	TRUE	previous	not_process	not_process	1_billion
config565	elmo_vocabulary	TRUE	TRUE	middle	not_process	any_line	google
config566	elmo_vocabulary	TRUE	TRUE	middle	not_process	same_line	google
config567	elmo_vocabulary	TRUE	TRUE	middle	not_process	not_process	google
config568	elmo_vocabulary	TRUE	TRUE	middle	not_process	any_line	1_billion
config569	elmo_vocabulary	TRUE	TRUE	middle	not_process	same_line	1_billion
config570	elmo_vocabulary	TRUE	TRUE	middle	not_process	not_process	1_billion
config571	elmo_vocabulary	TRUE	TRUE	all	not_process	any_line	google
config572	elmo_vocabulary	TRUE	TRUE	all	not_process	same_line	google
config573	elmo_vocabulary	TRUE	TRUE	all	not_process	not_process	google
config574	elmo_vocabulary	TRUE	TRUE	all	not_process	any_line	1_billion
config575	elmo_vocabulary	TRUE	TRUE	all	not_process	same_line	1_billion
config576	elmo_vocabulary	TRUE	TRUE	all	not_process	not_process	1_billion

10.5. Resultado configuraciones modelo de lenguaje

Tabla 10.2: Cantidad de victorias por configuración ordenadas de forma descendiente de la prueba en la Subsección 5.3.1.

Configuración	Cantidad victorias
config20	16
config19	14
config21	12
config298	12
config299	12
config300	12
config310	12
config311	12
config312	12
config7	11
config68	11
config309	11
config8	10
config9	10
config10	10
config11	10
config12	10
config13	10
config69	10
config296	10
config301	10
config346	10
config347	10
config348	10
config358	10
config359	10
config360	10
config22	9
config23	9
config24	9
config289	9
config290	9
config291	9
config302	9
config307	9
config308	9
config338	9
config339	9
config14	8
config15	8

config16	8
config17	8
config18	8
config55	8
config56	8
config58	8
config59	8
config60	8
config67	8
config70	8
config71	8
config72	8
config303	8
config355	8
config356	8
config357	8
config4	7
config63	7
config64	7
config65	7
config66	7
config292	7
config293	7
config294	7
config295	7
config297	7
config304	7
config305	7
config306	7
config340	7
config341	7
config342	7
config350	7
config351	7
config352	7
config353	7
config354	7
config1	6
config2	6
config3	6
config5	6
config6	6
config38	6
config39	6
config49	6
config51	6

config52	6
config53	6
config54	6
config57	6
config61	6
config62	6
config334	6
config335	6
config336	6
config343	6
config344	6
config26	5
config27	5
config31	5
config33	5
config34	5
config35	5
config36	5
config40	5
config41	5
config42	5
config44	5
config45	5
config46	5
config47	5
config48	5
config50	5
config313	5
config314	5
config322	5
config323	5
config324	5
config332	5
config337	5
config345	5
config349	5
config25	4
config28	4
config29	4
config30	4
config32	4
config37	4
config43	4
config157	4
config315	4
config316	4

config317	4
config318	4
config319	4
config320	4
config321	4
config327	4
config328	4
config329	4
config330	4
config91	3
config92	3
config146	3
config163	3
config190	3
config191	3
config192	3
config205	3
config206	3
config325	3
config326	3
config331	3
config333	3
config379	3
config445	3
config446	3
config93	2
config128	2
config134	2
config139	2
config140	2
config141	2
config151	2
config158	2
config159	2
config164	2
config165	2
config169	2
config175	2
config176	2
config178	2
config179	2
config180	2
config181	2
config184	2
config185	2
config186	2

config187	2
config188	2
config193	2
config194	2
config195	2
config199	2
config200	2
config380	2
config404	2
config427	2
config433	2
config434	2
config435	2
config436	2
config437	2
config438	2
config451	2
config452	2
config453	2
config460	2
config461	2
config462	2
config477	2
config478	2
config479	2
config480	2
config484	2
config485	2
config486	2
config499	2
config500	2
config73	1
config79	1
config80	1
config81	1
config85	1
config86	1
config87	1
config99	1
config103	1
config104	1
config105	1
config109	1
config110	1
config111	1
config115	1

config116	1
config117	1
config121	1
config123	1
config127	1
config129	1
config133	1
config135	1
config145	1
config147	1
config148	1
config149	1
config150	1
config152	1
config153	1
config154	1
config155	1
config156	1
config160	1
config161	1
config162	1
config166	1
config167	1
config168	1
config172	1
config173	1
config174	1
config177	1
config182	1
config183	1
config189	1
config196	1
config197	1
config198	1
config201	1
config202	1
config203	1
config204	1
config208	1
config209	1
config210	1
config214	1
config215	1
config216	1
config217	1
config231	1

config243	1
config267	1
config362	1
config364	1
config365	1
config366	1
config367	1
config368	1
config369	1
config373	1
config374	1
config375	1
config381	1
config385	1
config391	1
config392	1
config393	1
config397	1
config398	1
config399	1
config403	1
config405	1
config410	1
config411	1
config415	1
config416	1
config417	1
config421	1
config422	1
config423	1
config428	1
config429	1
config439	1
config440	1
config441	1
config442	1
config443	1
config444	1
config447	1
config448	1
config449	1
config450	1
config454	1
config455	1
config456	1
config464	1

config466	1
config467	1
config468	1
config471	1
config472	1
config473	1
config474	1
config481	1
config488	1
config489	1
config490	1
config491	1
config492	1
config494	1
config496	1
config497	1
config498	1
config501	1
config502	1
config503	1
config504	1
config74	0
config75	0
config76	0
config77	0
config78	0
config82	0
config83	0
config84	0
config88	0
config89	0
config90	0
config94	0
config95	0
config96	0
config97	0
config98	0
config100	0
config101	0
config102	0
config106	0
config107	0
config108	0
config112	0
config113	0
config114	0

config118	0
config119	0
config120	0
config122	0
config124	0
config125	0
config126	0
config130	0
config131	0
config132	0
config136	0
config137	0
config138	0
config142	0
config143	0
config144	0
config170	0
config171	0
config207	0
config211	0
config212	0
config213	0
config218	0
config219	0
config220	0
config221	0
config222	0
config223	0
config224	0
config225	0
config226	0
config227	0
config228	0
config229	0
config230	0
config232	0
config233	0
config234	0
config235	0
config236	0
config237	0
config238	0
config239	0
config240	0
config241	0
config242	0

config244	0
config245	0
config246	0
config247	0
config248	0
config249	0
config250	0
config251	0
config252	0
config253	0
config254	0
config255	0
config256	0
config257	0
config258	0
config259	0
config260	0
config261	0
config262	0
config263	0
config264	0
config265	0
config266	0
config268	0
config269	0
config270	0
config271	0
config272	0
config273	0
config274	0
config275	0
config276	0
config277	0
config278	0
config279	0
config280	0
config281	0
config282	0
config283	0
config284	0
config285	0
config286	0
config287	0
config288	0
config361	0
config363	0

config370	0
config371	0
config372	0
config376	0
config377	0
config378	0
config382	0
config383	0
config384	0
config386	0
config387	0
config388	0
config389	0
config390	0
config394	0
config395	0
config396	0
config400	0
config401	0
config402	0
config406	0
config407	0
config408	0
config409	0
config412	0
config413	0
config414	0
config418	0
config419	0
config420	0
config424	0
config425	0
config426	0
config430	0
config431	0
config432	0
config457	0
config458	0
config459	0
config463	0
config465	0
config469	0
config470	0
config475	0
config476	0
config482	0

config483	0
config487	0
config493	0
config495	0
config505	0
config506	0
config507	0
config508	0
config509	0
config510	0
config511	0
config512	0
config513	0
config514	0
config515	0
config516	0
config517	0
config518	0
config519	0
config520	0
config521	0
config522	0
config523	0
config524	0
config525	0
config526	0
config527	0
config528	0
config529	0
config530	0
config531	0
config532	0
config533	0
config534	0
config535	0
config536	0
config537	0
config538	0
config539	0
config540	0
config541	0
config542	0
config543	0
config544	0
config545	0
config546	0

config547	0
config548	0
config549	0
config550	0
config551	0
config552	0
config553	0
config554	0
config555	0
config556	0
config557	0
config558	0
config559	0
config560	0
config561	0
config562	0
config563	0
config564	0
config565	0
config566	0
config567	0
config568	0
config569	0
config570	0
config571	0
config572	0
config573	0
config574	0
config575	0
config576	0

Tabla 10.3: Promedio de puntaje respecto a los 100 documentos de la prueba en la Subsección 5.3.1 para cada configuración.

Configuración	Promedio
config67	0.7668453570945141705
config20	0.7668276747913389356
config61	0.7667074699151101527
config62	0.7666834002380204957
config57	0.7665723831619586104
config307	0.7663974552022276520
config308	0.7663841735985021280
config303	0.7662467860132241806
config309	0.7659515426033906740
config295	0.7658877708173926408
config355	0.7658721843066623500
config301	0.7658648270607663300
config356	0.7658588978845213780
config297	0.7658575900505821818
config302	0.7658515472858252260
config357	0.7658420047922329810
config350	0.7657180018776596426
config351	0.7657011028140152906
config296	0.7654611940913452623
config298	0.7654162270936826498
config299	0.7654029406905987628
config19	0.7653914357674810619
config300	0.7653860557330547558
config343	0.7653501440301032213
config68	0.7653446151050954414
config344	0.7653368564031867533
config21	0.7653205994024771209
config69	0.7653078443828348914
config349	0.7652970176780978350
config13	0.7652943140300138854
config14	0.7652702535241320954
config15	0.7652192291037460284
config7	0.7651919501610327964
config8	0.7651688673821279894
config63	0.7651654363520745307
config55	0.7651601594041022757
config56	0.7651370349697127747
config9	0.7651281532026329674
config310	0.7649591453218491852
config311	0.7649458600216159362
config312	0.7649289784961037992
config345	0.7648845972685275559
config346	0.7647925625775610366

config347	0.7647792717965867276
config348	0.7647623868390427206
config50	0.7643665835284622274
config2	0.7643402182863210444
config51	0.7643279260096458174
config358	0.7643261046307461141
config359	0.7643128149531405771
config360	0.7642959334276284401
config43	0.7642123452585945272
config38	0.7640945048407490961
config39	0.7640434612719231571
config31	0.7640162799667751498
config289	0.7639576830060083371
config33	0.7639480804119667628
config290	0.7639477593333063501
config291	0.7639341524723451951
config292	0.7638108478074203050
config293	0.7638009299227679150
config294	0.7637840846243849230
config304	0.7637018663712532492
config305	0.7636862719780176012
config306	0.7636693904525054642
config10	0.7636434680690223605
config22	0.7636354194405223960
config11	0.7636158009040217315
config4	0.7636104453730312420
config23	0.7636093637636411500
config5	0.7635971635083513050
config52	0.7635755185217220003
config12	0.7635656777773829805
config53	0.7635622276620643433
config24	0.7635592446710933970
config6	0.7635588811717474340
config54	0.7635295020572116463
config332	0.7635120354909793836
config338	0.7634761119278945059
config70	0.7634731825527224104
config58	0.7634702080145489566
config339	0.7634625050669333509
config71	0.7634470848923966454
config59	0.7634425004636797026
config72	0.7634020362326163974
config60	0.7633974437936452146
config327	0.7633578066718601966
config331	0.7633214494125769539
config340	0.7632954108752283580

config333	0.7632911801044242659
config341	0.7632855109861365950
config342	0.7632686656877536030
config325	0.7631877721069718717
config326	0.7631744265515466137
config337	0.7631340296354695207
config352	0.7631013738694539561
config353	0.7630857766011884521
config354	0.7630688950756763151
config44	0.7630420077993930185
config45	0.7629996834466705675
config319	0.7629938606313754220
config16	0.7629873996112107694
config37	0.7629855359675843237
config320	0.7629805131255942390
config321	0.7629635788108098700
config17	0.7629632913722005774
config18	0.7629219457862213074
config32	0.7628459585072343687
config64	0.7628344954464803664
config65	0.7628103492296839414
config66	0.7627662042041687074
config49	0.7627579503973006995
config1	0.7627466964275246831
config3	0.7626877551511751061
config322	0.7625819627570962989
config323	0.7625686104968419829
config324	0.7625516903536605389
config451	0.7623529170573100716
config452	0.7623396267547452726
config453	0.7623215586929274626
config445	0.7622277872531780359
config446	0.7622144958503478079
config157	0.7621744633502578005
config334	0.7621281566480416329
config335	0.7621148055031746999
config336	0.7620978888022913019
config205	0.7620648640920552925
config206	0.7620405857817566855
config151	0.7620110794820357707
config447	0.7617768323133258186
config501	0.7617452899084398466
config26	0.7617274023201250188
config27	0.7616932651696683758
config494	0.7616179564285916899
config439	0.7613479631353180221

config28	0.7613439953928527167
config442	0.7613395464558796339
config440	0.7613346756568210361
config499	0.7613334175697737104
config29	0.7613263319765723177
config443	0.7613262513612043629
config500	0.7613201253611379164
config441	0.7613166053454858101
config34	0.7613120384029042093
config444	0.7613081852388878039
config46	0.7612970265831877166
config30	0.7612930959232378757
config35	0.7612798908233891723
config47	0.7612664899715218726
config36	0.7612347667958306153
config48	0.7612213692768537706
config493	0.7611917077906988390
config488	0.7611676586790086913
config495	0.7611603408515887060
config489	0.7611495834227101513
config313	0.7610043544738711330
config314	0.7609944074196350980
config163	0.7609107381177572434
config164	0.7608820904624277284
config328	0.7608810859859099800
config454	0.7608808115368804335
config455	0.7608675175455739345
config329	0.7608654236294372370
config456	0.7608494548556980665
config330	0.7608485069285538390
config316	0.7608454869404658831
config315	0.7608424101016179434
config165	0.7608385795258387414
config317	0.7608355472025166981
config211	0.7608343590434815624
config318	0.7608186681622346741
config212	0.7608056855677756174
not_processed	0.7607944513612512902
config158	0.7607683700055045740
config213	0.7607677125580478154
config487	0.7607405138001211485
config159	0.7607161446051973040
config490	0.7606709414059305893
config40	0.7606655891926225269
config491	0.7606576419229438233
config492	0.7606395758006272643

config41	0.7606370017518620429
config207	0.7606043207497812435
config42	0.7606006883891526769
config152	0.7605983176615622764
config153	0.7605564045251475204
config199	0.7605327180646402499
config200	0.7605093970700807469
config201	0.7604656850765275189
config25	0.7604618498558284888
config502	0.7602031088665420138
config503	0.7601898104874425218
config504	0.7601717477975666538
config433	0.7598900365038103300
config434	0.7598801197089624130
config435	0.7598653776165431430
config448	0.7596946881445860287
config449	0.7596790897745050647
config450	0.7596610261877189947
config187	0.7596464493439054301
config436	0.7596218851711654573
config188	0.7596134060029906671
config437	0.7596119741716490223
config438	0.7595939410765145133
config193	0.7595771432771227690
config189	0.7595699211059208631
config194	0.7595625165435253340
config146	0.7595584436410743850
config181	0.7595361011132585805
config195	0.7595195367868039070
config481	0.7593974329575924373
config175	0.7593629205570171597
config176	0.7593352035555115647
config177	0.7592933086676579127
config477	0.7591565289915364783
config484	0.7590864541113779942
config485	0.7590765611104892662
config486	0.7590585280153547572
config496	0.7590493651858086878
config166	0.7590435676771569235
config154	0.7590376496767461159
config497	0.7590337639376122438
config167	0.7590173121826327225
config471	0.7590166537737859319
config498	0.7590157003508261738
config155	0.7590097820150727479
config482	0.7590092275262522796

config483	0.7589944899655795746
config475	0.7589790674367075967
config168	0.7589660040942282795
config476	0.7589657171103759857
config156	0.7589584677408257729
config469	0.7588428132220341609
config470	0.7588294617834693189
config93	0.7588259484358911995
config214	0.7588258151503225408
config202	0.7588069902500102020
config148	0.7588068102679038736
config215	0.7587995181308622128
config149	0.7587934503356648016
config203	0.7587790826622219870
config216	0.7587532804667818508
config196	0.7587525267735620687
config150	0.7587508436293963876
config197	0.7587391570189724147
config204	0.7587328348310916540
config198	0.7587021071791220427
config464	0.7585726319881888713
config182	0.7584321473798736392
config160	0.7583938215335917866
config183	0.7583798606103177042
config463	0.7583774017651205914
config161	0.7583695166373113676
config465	0.7583459472422693194
config162	0.7583269920725625486
config466	0.7582206878803631063
config467	0.7582073297361847673
config468	0.7581892298552349493
config208	0.7581853086544048755
config209	0.7581609656137107215
config210	0.7581156414611253805
config145	0.7580425440498488152
config147	0.7579792909970369832
config91	0.7578598015531077006
config92	0.7578311788308025436
config478	0.7577652508517521405
config479	0.7577518938234881815
config85	0.7577440037632795283
config480	0.7577337973844237265
config86	0.7577198073589510973
config87	0.7576644993496838623
config140	0.7576366977084999572
config79	0.7575155456266250629

config134	0.7574951595070126865
config80	0.7574923200447026049
config81	0.7574473269503245519
config128	0.7572889090936436467
config169	0.7569629671481948115
config379	0.7568250922112811514
config380	0.7568117910786308154
config178	0.7566952751154335377
config190	0.7566942888157122442
config191	0.7566635477248297792
config179	0.7566629223764479877
config381	0.7566546410294683750
config139	0.7566286911470824082
config192	0.7566172486439080592
config180	0.7566166191102495667
config374	0.7566015198996162804
config472	0.7565895094662939510
config473	0.7565738430925064040
config141	0.7565563909489318252
config474	0.7565557457565317470
config172	0.7565265090666344200
config173	0.7565087615094970290
config133	0.7564872172296003933
config457	0.7564797286191766785
config373	0.7564793741502095250
config174	0.7564712156328195530
config458	0.7564697923749419135
config459	0.7564550792806381495
config375	0.7564480140172556710
config135	0.7564132182740871593
config460	0.7563682080883990665
config461	0.7563582724228518005
config462	0.7563402069726075145
config370	0.7562693431954670744
config127	0.7562693225225421419
config371	0.7562560375952467774
config372	0.7562380872890867834
config115	0.7562372356826137855
config116	0.7562042189253317545
config129	0.7561992954251468269
config367	0.7561966253380718239
config368	0.7561833229975030009
config369	0.7561652583729459789
config117	0.7561549453875226875
config184	0.7560562385731119587
config111	0.7560402646944775082

config185	0.7560274491819933787
config186	0.7559899674118016047
config103	0.7558984323834744047
config105	0.7558257303173440287
config382	0.7558142654803737975
config427	0.7558119993042548164
config383	0.7558009609871594655
config384	0.7557830169260693295
config170	0.7557211801879250997
config171	0.7556827330669489527
config428	0.7556513690766288949
config429	0.7556333165574472739
config421	0.7555863114427595435
config422	0.7555730028328499895
config423	0.7555549359578795845
config415	0.7551697384900307545
config416	0.7551564286669343045
config418	0.7551539207164715768
config419	0.7551406091896098378
config417	0.7551383640423772825
config420	0.7551226588834498438
config109	0.7551013905902730017
config110	0.7550727273233606717
config104	0.7548371494936382633
config404	0.7547596337720820202
config430	0.7546833520859814245
config431	0.7546700416666467165
config432	0.7546520976055565805
config403	0.7546110821660364679
config376	0.7546099506493573112
config377	0.7545943444548633262
config82	0.7545887561606847361
config405	0.7545795944550478409
config378	0.7545763958910376982
config94	0.7545686386130104951
config83	0.7545609655373318681
config95	0.7545424564155583071
config84	0.7545105426528098611
config96	0.7544946596544653821
config397	0.7543944976044807359
config398	0.7543811334406190149
config399	0.7543630003946109469
config394	0.7542784822944467493
config395	0.7542651106372754743
config396	0.7542470902027435573
config391	0.7541317237600461685

config392	0.7539592964046609080
config393	0.7539411656075598270
config406	0.7538184163060368168
config407	0.7538050457684793738
config408	0.7537870316343265068
config362	0.7536409554671912498
config424	0.7534843837076769250
config425	0.7534687731006626410
config426	0.7534508245368370130
config73	0.7533739855739519087
config130	0.7532414375236972229
config142	0.7532392603076364298
config131	0.7532135790207143589
config143	0.7532130066213232918
config361	0.7532088838683054080
config363	0.7531844713781197960
config144	0.7531702967808830348
config132	0.7531682327022987269
config364	0.7529488227454963547
config365	0.7529389039606114017
config366	0.7529211028957383107
config76	0.7528491585695277687
config77	0.7528358714832687057
config78	0.7527898188478403247
config88	0.7527758756788873443
config89	0.7527516288944190343
config410	0.7527145330120791835
config90	0.7527125903211574803
config411	0.7527000341302847105
config400	0.7526059046784444800
config401	0.7525902303978019650
config402	0.7525722117332665470
config409	0.7522747068088099690
config121	0.7521942907695632800
config123	0.7521358085737821800
config74	0.7520496954388886520
config75	0.7520001215323121250
config106	0.7519372602524598468
config118	0.7519202713410709398
config412	0.7519165426463140878
config413	0.7519066425516815198
config107	0.7519049815466777198
config119	0.7518896004979343988
config414	0.7518888414868084288
config108	0.7518544149085921258
config120	0.7518416721707055048

config124	0.7516330981903058768
config125	0.7516197885124738288
config126	0.7515793813673614288
config385	0.7515009576184699517
config136	0.7514519183097535986
config137	0.7514276041619682016
config138	0.7513857363384147896
config386	0.7510245275755538644
config387	0.7510099878707718484
config122	0.7508697276379709834
config388	0.7507816391274893941
config389	0.7507716984826971841
config390	0.7507538290870535351
config99	0.7506608937590875446
config112	0.7503406354344753217
config113	0.7503119011437905507
config114	0.7502727712710254387
config100	0.7502423841964790520
config101	0.7502247191532286220
config102	0.7501785759156956620
config97	0.7494303625030784866
config98	0.7494113575832440746
config523	0.7183680348211275572
config524	0.7183546285359833432
config517	0.7181606885162263936
config525	0.7181458345271593922
config518	0.7180176361387385368
config519	0.7179404934633029356
config511	0.7179259506439221180
config512	0.7179125430992050200
config513	0.7175718910466665445
config514	0.7175514268916276633
config515	0.7175380240307950073
config516	0.7173314084855403103
config520	0.7171389568155340942
config521	0.7171233380419572922
config235	0.7171137297785596418
config236	0.7170845177259746308
config526	0.7170801724335594438
config527	0.7170667695727267888
config229	0.7170036874380480346
config230	0.7169784969594573436
config522	0.7169166280009017322
config237	0.7168708068545190168
config528	0.7168601602044000458
config223	0.7167905000254057127

config224	0.7167662483225386177
config225	0.7165587389324657917
config231	0.7163968789712995482
config547	0.7163530137895031095
config548	0.7163395464116087995
config541	0.7162707987718982628
config549	0.7161302821934842115
config542	0.7161044875509689861
config543	0.7160492848238544028
config571	0.7159411154977617480
config535	0.7158995366190622318
config536	0.7158860679561048018
config572	0.7157855498361812727
config538	0.7157644786278497281
config539	0.7157510116697933191
config565	0.7157297198556155085
config573	0.7157188401764528750
config537	0.7156787662346628708
config566	0.7155778698145147181
config540	0.7155431098276393921
config559	0.7154986492876045115
config567	0.7153715919853538241
config560	0.7153441418010987342
config544	0.7153248682312049794
config545	0.7153091812646327654
config550	0.7152846436972133438
config561	0.7152781269689209325
config551	0.7152711767391569358
config546	0.7151011835431817904
config552	0.7150632811426592358
config562	0.7150467285778608075
config563	0.7150333196854377045
config508	0.7149840208998874353
config509	0.7149742283565353153
config506	0.7148373251402323604
config564	0.7148265515376443715
config510	0.7147575960674631973
config505	0.7147371285407387604
config260	0.7147116709066855403
config507	0.7146221217223951314
config568	0.7146110717440807166
config254	0.7146110404948309748
config569	0.7145954485492453526
config574	0.7145589854897988011
config575	0.7145455765973757001
config261	0.7144968214878544298

config283	0.7144802979022700436
config226	0.7144531076975542389
config284	0.7144510313010303316
config238	0.7144364798442821948
config227	0.7144243701266206519
config247	0.7144211507084139550
config239	0.7144093442597246388
config255	0.7143940920975538328
config248	0.7143924170036595370
config570	0.7143885857401922796
config259	0.7143694348026648771
config576	0.7143388146265103211
config277	0.7143326183545688802
config253	0.7142601936915730337
config285	0.7142432673275069616
config228	0.7142262187318420013
config240	0.7142158347586372882
config279	0.7140973368607093952
config278	0.7139332389193801562
config273	0.7138890855812134950
config249	0.7138182499495654789
config271	0.7137623150118106207
config272	0.7137380227603836227
config232	0.7134039085651304575
config233	0.7133786457973152185
config234	0.7131958310569745375
config220	0.7129918215119638614
config221	0.7129783813940478614
config532	0.7129261137281965982
config218	0.7129251028365676347
config533	0.7129162977390991642
config222	0.7127660724363817124
config219	0.7127093095833686472
config534	0.7126983701994734262
config217	0.7126948577474726098
config529	0.7126733392035205456
config530	0.7126635244404837306
config556	0.7125475894526912378
config557	0.7125378156147050748
config554	0.7125133818991012262
config531	0.7124478979839194286
config553	0.7124049364685956171
config558	0.7123212546170118448
config555	0.7121811858561910041
config250	0.7120739194231345410
config262	0.7120522650330298419

config251	0.7120406742162460140
config263	0.7120206313666577269
config252	0.7118347455771600513
config264	0.7118193530436687832
config274	0.7116971106195610054
config286	0.7116952774124638372
config275	0.7116683061541844464
config287	0.7116680736040268922
config288	0.7114797556481041839
config276	0.7114754262329399671
config256	0.7112297007447720067
config257	0.7111999355370349007
config258	0.7110093876982374390
config280	0.7106814818507330944
config281	0.7106561493835354364
config244	0.7106372725985687782
config245	0.7106194476290426102
config241	0.7105108722075221849
config242	0.7104922149064822449
config282	0.7104704621942518944
config246	0.7104060195590148832
config265	0.7103717358420877269
config266	0.7103574497443145889
config268	0.7103566933320478484
config269	0.7103432345510014344
config270	0.7101369275479876718
config243	0.7100100982543131585
config267	0.7099037057399919853

Tabla 10.4: Cantidad de documentos para los que se incrementó o decrementó el puntaje de similitud de la prueba en la Subsección 5.3.1.

Configuración	Cantidad docs con incremento	Cantidad docs con decremento
config19	82	18
config20	82	18
config21	82	18
config8	81	19
config9	81	19
config307	81	19
config308	81	19
config68	81	19
config301	81	19
config13	81	19
config69	81	19
config302	81	19
config14	81	19
config303	81	19
config7	81	19
config15	81	19
config297	80	20
config10	80	20
config11	80	20
config12	80	20
config309	80	20
config349	80	20
config295	80	20
config63	80	20
config24	79	21
config355	79	21
config67	79	21
config356	79	21
config357	79	21
config61	79	21
config350	79	21
config22	79	21
config62	79	21
config351	79	21
config23	79	21
config55	79	21
config296	78	22
config344	78	22
config56	78	22
config57	78	22
config298	78	22
config299	78	22

config300	78	22
config343	78	22
config58	77	23
config59	77	23
config60	77	23
config304	76	24
config312	76	24
config305	76	24
config345	76	24
config306	76	24
config346	76	24
config2	76	24
config347	76	24
config348	76	24
config310	76	24
config70	76	24
config311	76	24
config71	76	24
config360	75	25
config16	75	25
config72	75	25
config289	75	25
config1	75	25
config17	75	25
config49	75	25
config290	75	25
config18	75	25
config291	75	25
config3	75	25
config358	75	25
config359	75	25
config352	74	26
config353	74	26
config338	74	26
config354	74	26
config339	74	26
config50	73	27
config66	73	27
config51	73	27
config292	73	27
config4	73	27
config52	73	27
config293	73	27
config5	73	27
config53	73	27
config294	73	27

config6	73	27
config54	73	27
config64	72	28
config65	72	28
config332	72	28
config327	72	28
config32	71	29
config337	71	29
config43	71	29
config340	71	29
config44	71	29
config341	71	29
config45	71	29
config342	71	29
config31	71	29
config33	70	30
config37	70	30
config328	69	31
config329	69	31
config330	69	31
config331	69	31
config325	69	31
config333	69	31
config326	69	31
config38	69	31
config39	69	31
config320	68	32
config321	68	32
config322	68	32
config34	68	32
config323	68	32
config324	68	32
config319	68	32
config336	67	33
config334	67	33
config46	67	33
config335	67	33
config48	66	34
config35	66	34
config36	66	34
config47	66	34
config313	64	36
config314	64	36
config451	64	36
config452	64	36
config445	64	36

config453	64	36
config446	64	36
config25	63	37
config442	63	37
config443	63	37
config444	63	37
config447	63	37
config40	62	38
config315	62	38
config499	62	38
config316	62	38
config500	62	38
config28	62	38
config317	62	38
config501	62	38
config29	62	38
config318	62	38
config30	62	38
config440	61	39
config488	61	39
config441	61	39
config489	61	39
config41	61	39
config26	61	39
config27	61	39
config439	61	39
config487	61	39
config448	60	40
config456	60	40
config449	60	40
config450	60	40
config42	60	40
config454	60	40
config455	60	40
config205	59	41
config493	59	41
config206	59	41
config494	59	41
config495	59	41
config200	58	42
config496	58	42
config433	58	42
config481	58	42
config497	58	42
config434	58	42
config490	58	42

config498	58	42
config163	58	42
config211	58	42
config435	58	42
config491	58	42
config212	58	42
config492	58	42
config157	58	42
config199	58	42
config152	57	43
config153	57	43
config201	57	43
config164	57	43
config165	57	43
config213	57	43
config158	57	43
config151	57	43
config504	56	44
config477	56	44
config502	56	44
config159	56	44
config207	56	44
config503	56	44
config168	55	45
config154	55	45
config155	55	45
config156	55	45
config166	55	45
config167	55	45
config471	55	45
config160	54	46
config161	54	46
config162	54	46
config202	54	46
config482	54	46
config203	54	46
config475	54	46
config483	54	46
config436	54	46
config476	54	46
config437	54	46
config438	54	46
config216	53	47
config465	53	47
config204	53	47
config469	53	47

config214	53	47
config470	53	47
config175	53	47
config215	53	47
config463	53	47
config176	52	48
config464	52	48
config177	52	48
config193	52	48
config194	52	48
config195	52	48
config484	52	48
config485	52	48
config182	52	48
config486	52	48
config183	52	48
config208	51	49
config209	51	49
config146	51	49
config178	51	49
config210	51	49
config187	51	49
config148	51	49
config188	51	49
config149	51	49
config181	51	49
config189	51	49
config150	51	49
config192	50	50
config466	50	50
config179	50	50
config467	50	50
config180	50	50
config468	50	50
config190	50	50
config191	50	50
config472	49	51
config480	49	51
config145	49	51
config147	49	51
config478	49	51
config479	49	51
config184	48	52
config185	48	52
config473	48	52
config186	48	52

config474	48	52
config196	48	52
config197	48	52
config93	48	52
config198	48	52
config169	47	53
config80	46	54
config172	46	54
config173	46	54
config174	46	54
config79	46	54
config81	45	55
config170	45	55
config171	45	55
config91	45	55
config140	45	55
config460	45	55
config92	45	55
config461	45	55
config85	45	55
config462	45	55
config86	45	55
config87	45	55
config128	44	56
config376	44	56
config129	44	56
config377	44	56
config457	44	56
config378	44	56
config458	44	56
config139	44	56
config459	44	56
config115	44	56
config116	44	56
config133	44	56
config141	44	56
config109	44	56
config117	44	56
config134	44	56
config374	44	56
config135	44	56
config127	44	56
config368	43	57
config369	43	57
config370	43	57
config371	43	57

config379	43	57
config372	43	57
config380	43	57
config373	43	57
config110	43	57
config367	43	57
config375	43	57
config384	42	58
config104	42	58
config105	42	58
config382	42	58
config383	42	58
config103	42	58
config111	42	58
config73	41	59
config381	41	59
config94	41	59
config408	40	60
config96	40	60
config361	40	60
config362	40	60
config394	40	60
config82	40	60
config363	40	60
config395	40	60
config427	40	60
config83	40	60
config396	40	60
config404	40	60
config406	40	60
config407	40	60
config95	40	60
config418	39	61
config74	39	61
config403	39	61
config419	39	61
config75	39	61
config420	39	61
config428	39	61
config84	39	61
config405	39	61
config421	39	61
config429	39	61
config422	39	61
config423	39	61
config144	38	62

config400	38	62
config416	38	62
config424	38	62
config432	38	62
config401	38	62
config417	38	62
config425	38	62
config402	38	62
config426	38	62
config76	38	62
config397	38	62
config77	38	62
config398	38	62
config430	38	62
config78	38	62
config391	38	62
config399	38	62
config415	38	62
config431	38	62
config130	37	63
config131	37	63
config132	37	63
config364	37	63
config365	37	63
config142	37	63
config366	37	63
config143	37	63
config392	36	64
config88	36	64
config393	36	64
config89	36	64
config121	36	64
config90	36	64
config123	36	64
config106	35	65
config107	35	65
config108	35	65
config120	34	66
config385	34	66
config122	34	66
config118	34	66
config119	34	66
config409	33	67
config97	33	67
config410	33	67
config98	33	67

config411	33	67
config412	33	67
config413	33	67
config414	33	67
config99	32	68
config100	32	68
config124	32	68
config101	32	68
config136	31	69
config137	31	69
config138	31	69
config386	31	69
config387	31	69
config125	31	69
config102	31	69
config126	31	69
config112	30	70
config113	30	70
config114	30	70
config388	27	73
config389	27	73
config390	27	73
config224	24	76
config225	24	76
config235	24	76
config236	24	76
config229	24	76
config237	24	76
config230	24	76
config223	24	76
config231	24	76
config523	22	78
config524	22	78
config517	22	78
config525	22	78
config519	22	78
config248	21	79
config272	21	79
config512	21	79
config249	21	79
config273	21	79
config513	21	79
config259	21	79
config283	21	79
config284	21	79
config253	21	79

config285	21	79
config254	21	79
config247	21	79
config255	21	79
config271	21	79
config511	21	79
config260	20	80
config261	20	80
config277	20	80
config278	20	80
config518	20	80
config279	20	80
config528	19	81
config217	19	81
config514	19	81
config515	19	81
config516	19	81
config526	19	81
config527	19	81
config240	18	82
config520	18	82
config536	18	82
config560	18	82
config521	18	82
config537	18	82
config561	18	82
config218	18	82
config226	18	82
config522	18	82
config219	18	82
config227	18	82
config571	18	82
config228	18	82
config572	18	82
config541	18	82
config565	18	82
config573	18	82
config238	18	82
config542	18	82
config566	18	82
config239	18	82
config535	18	82
config543	18	82
config559	18	82
config567	18	82
config576	17	83

config547	17	83
config220	17	83
config548	17	83
config221	17	83
config549	17	83
config222	17	83
config574	17	83
config575	17	83
config232	16	84
config288	16	84
config552	16	84
config568	16	84
config233	16	84
config241	16	84
config505	16	84
config569	16	84
config234	16	84
config242	16	84
config274	16	84
config538	16	84
config562	16	84
config570	16	84
config243	16	84
config275	16	84
config539	16	84
config563	16	84
config276	16	84
config540	16	84
config564	16	84
config286	16	84
config550	16	84
config287	16	84
config551	16	84
config264	15	85
config280	15	85
config544	15	85
config265	15	85
config281	15	85
config545	15	85
config250	15	85
config266	15	85
config282	15	85
config506	15	85
config546	15	85
config251	15	85
config267	15	85

config507	15	85
config252	15	85
config262	15	85
config263	15	85
config256	14	86
config257	14	86
config553	14	86
config258	14	86
config554	14	86
config555	14	86
config244	14	86
config508	14	86
config556	14	86
config245	14	86
config509	14	86
config557	14	86
config246	14	86
config510	14	86
config558	14	86
config529	13	87
config530	13	87
config531	13	87
config268	13	87
config269	13	87
config270	13	87
config532	11	89
config533	11	89
config534	11	89
not_processed	0	0

Tabla 10.5: Percentil 75 de cada configuración en los 100 documentos de la prueba en la Subsección 5.3.1.

Configuración	Percentil 75
config67	0.876118391
config19	0.875851758
config20	0.875750486
config21	0.875477602
config301	0.875287663
config302	0.875287663
config296	0.87525621
config13	0.875137363
config14	0.875137363
config309	0.875126643
config7	0.875109685
config8	0.875109685
config15	0.874915298
config9	0.874753849
config61	0.874714882
config62	0.874714882
config57	0.873879527
config68	0.873793994
config69	0.873793994
config2	0.873585237
config1	0.873501194
config22	0.873297453
config23	0.873297453
config3	0.8732172
config44	0.873177988
config43	0.873021111
config63	0.873004918
config303	0.872693949
config55	0.872687971
config56	0.872687971
config4	0.872686575
config5	0.872686575
config352	0.872678687
config353	0.872678687
config354	0.872678687
config307	0.872640301
config308	0.872640301
config45	0.872593646
config10	0.872560393
config11	0.872560393
config345	0.87252081
config24	0.872513229
config31	0.872488593

config38	0.872483895
config6	0.872444804
config349	0.872424168
config50	0.872310372
config51	0.872310372
config325	0.872291029
config326	0.872291029
config337	0.872219143
config331	0.872129808
config333	0.872129808
config350	0.872039894
config351	0.872039894
config355	0.871999556
config356	0.871999556
config357	0.871999556
config343	0.871986315
config344	0.871986315
config33	0.871945856
config358	0.871884968
config359	0.871884968
config360	0.871884968
config37	0.871878494
config16	0.871874749
config17	0.871874749
config346	0.871841158
config347	0.871841158
config348	0.871841158
config304	0.871821802
config305	0.871821802
config306	0.871821802
config39	0.871817468
config32	0.871782979
config338	0.871644852
config339	0.871644852
config295	0.871595068
config297	0.871595068
config12	0.871438666
config310	0.871046787
config311	0.871046787
config312	0.871046787
config49	0.871001116
config298	0.870973994
config299	0.870973994
config300	0.870973994
config289	0.87094032
config290	0.87094032

config291	0.87094032
config315	0.870880554
config340	0.870836884
config341	0.870836884
config342	0.870836884
config18	0.870753165
config70	0.870361601
config71	0.870361601
config72	0.870361601
config327	0.870188386
config332	0.870134672
config26	0.870000513
config25	0.869996185
config46	0.869971376
config47	0.869971376
config319	0.869844115
config320	0.869844115
config321	0.869844115
config34	0.869701207
config35	0.869701207
config292	0.869663847
config293	0.869663847
config294	0.869663847
config48	0.869595887
config40	0.86958156
config41	0.86958156
config28	0.869466466
config29	0.869466466
config27	0.869333189
config36	0.869325742
config30	0.869293886
config334	0.869187462
config335	0.869187462
config336	0.869187462
config52	0.869175608
config53	0.869175608
config54	0.869175608
config328	0.868992727
config329	0.868992727
config330	0.868992727
config58	0.868751621
config59	0.868751621
config60	0.868751621
config313	0.868531223
config314	0.868531223
config42	0.868455174

config64	0.868335985
config65	0.868335985
config66	0.868335985
config322	0.86814163
config323	0.86814163
config324	0.86814163
config157	0.867387496
config151	0.867329191
config205	0.86721323
config206	0.86721323
config79	0.867200037
config85	0.867195789
config86	0.867182186
config163	0.867180904
config164	0.867104863
config165	0.867104863
config80	0.867082183
config152	0.8670263
config158	0.867014365
config159	0.867014365
config153	0.866955997
config316	0.866951191
config317	0.866951191
config318	0.866951191
config91	0.86683729
config92	0.866823688
config81	0.866808893
config211	0.866727644
config199	0.866669322
config212	0.866651444
config447	0.866619779
config200	0.866593127
config201	0.866552771
config213	0.866552771
config207	0.86648712
config499	0.866254953
config500	0.866254953
config493	0.866198393
config495	0.866198393
config87	0.86617641
config182	0.866011416
config183	0.866011416
config439	0.865968108
config440	0.865968108
config441	0.865968108
config93	0.865884517

config181	0.865824111
config187	0.865824111
config487	0.865808805
config166	0.86570203
config175	0.865648913
config475	0.865616146
config476	0.865616146
config147	0.865602581
config188	0.865594217
config189	0.865594217
config145	0.865584653
config148	0.865576934
config149	0.865576934
config150	0.865576934
config469	0.865547138
config470	0.865547138
config154	0.865523544
config160	0.865523544
config167	0.865473409
config168	0.865473409
config196	0.86546497
config197	0.86546497
config146	0.865460536
config176	0.865419034
config177	0.865419034
config381	0.865399411
config155	0.865294924
config156	0.865294924
config161	0.865294924
config162	0.865294924
config482	0.865281925
config483	0.865281925
config194	0.865059466
config195	0.865059466
config193	0.865005175
config214	0.864973652
config454	0.864937926
config455	0.864937926
config456	0.864937926
config202	0.864911775
config463	0.864894407
config465	0.864894407
config373	0.864881283
config375	0.864881283
config215	0.864744785
config203	0.864682908

config457	0.864620202
config458	0.864620202
config459	0.864620202
config198	0.864575416
config134	0.864435995
config445	0.864433248
config446	0.864433248
config374	0.864384801
config109	0.864352826
config451	0.864342757
config452	0.864342757
config453	0.864342757
config128	0.864332264
config379	0.864326953
config380	0.864326953
config94	0.864323184
config95	0.864246849
config442	0.864215181
config443	0.864215181
config444	0.864215181
config103	0.864175405
config110	0.86412242
config216	0.864079316
config367	0.864008342
config368	0.864008342
config369	0.864008342
config104	0.863947201
config208	0.863925155
config204	0.863893482
config209	0.863848866
config82	0.863810348
config170	0.86374648
config171	0.86374648
config83	0.863734013
config172	0.863684671
config127	0.863683537
config129	0.863683537
config140	0.863667303
config478	0.86365934
config479	0.86365934
config480	0.86365934
config111	0.863656667
config115	0.863651604
config105	0.863641392
config133	0.863637473
config135	0.863637473

config116	0.863618267
config178	0.863512457
config173	0.863510543
config174	0.863510543
config117	0.863453389
config169	0.863402895
config190	0.863381473
config184	0.863310978
config179	0.863282156
config180	0.863282156
config477	0.863221831
config433	0.86317166
config434	0.86317166
config435	0.86317166
config397	0.863153273
config398	0.863153273
config399	0.863153273
config471	0.863152959
config191	0.863151172
config192	0.863151172
config96	0.863124762
config496	0.863098687
config497	0.863098687
config498	0.863098687
config185	0.863080677
config186	0.863080677
config403	0.86305663
config405	0.86305663
config466	0.863018229
config467	0.863018229
config468	0.863018229
config502	0.862956682
config503	0.862956682
config504	0.862956682
config428	0.862943219
config429	0.862943219
config139	0.862942809
config141	0.862942809
config464	0.86287431
config490	0.862792753
config491	0.862792753
config492	0.862792753
config404	0.86275774
config210	0.862725891
config88	0.862642981
config84	0.862611793

config89	0.862566647
config361	0.86249747
config363	0.86249747
config501	0.862337828
config392	0.862251536
config393	0.862251536
config494	0.862249561
config382	0.862239112
config383	0.862239112
config384	0.862239112
config370	0.862218906
config371	0.862218906
config372	0.862218906
config481	0.862190337
config394	0.862182409
config395	0.862182409
config396	0.862182409
config488	0.862113166
config489	0.862113166
config448	0.861968472
config449	0.861968472
config450	0.861968472
config391	0.861962897
config73	0.861734507
config376	0.861714308
config377	0.861714308
config378	0.861714308
config406	0.861696431
config407	0.861696431
config408	0.861696431
config436	0.861660587
config437	0.861602897
config438	0.861602897
config106	0.861547524
config118	0.861485203
config90	0.861444436
config119	0.86138152
config107	0.861316724
config76	0.861283961
config77	0.861283961
config430	0.861252681
config431	0.861252681
config432	0.861252681
config484	0.861163064
config485	0.861163064
config486	0.861163064

config362	0.861154703
config120	0.861123453
config136	0.86108952
config137	0.861064111
config74	0.861042482
config75	0.861042482
config364	0.861038523
config365	0.861038523
config366	0.861038523
config78	0.861022335
config421	0.860907511
config422	0.860907511
config423	0.860907511
config123	0.860833268
config121	0.860814702
config386	0.860794291
config387	0.860794291
config427	0.860767029
config112	0.860747797
config138	0.860715422
config108	0.860670864
config113	0.860670864
config114	0.860670864
config409	0.860657547
config415	0.860616155
config416	0.860616155
config417	0.860616155
config418	0.860577901
config419	0.860577901
config420	0.860577901
config100	0.860544936
config424	0.860533537
config425	0.860533537
config426	0.860533537
config101	0.860486985
config102	0.860486985
config400	0.860305656
config401	0.860305656
config402	0.860305656
config410	0.860085296
config411	0.860085296
config472	0.860009799
config473	0.860009799
config474	0.860009799
config142	0.859751792
config143	0.859751792

config144	0.859751792
config122	0.859643279
config412	0.859636963
config98	0.859571713
config460	0.8595697
config99	0.859550334
config97	0.859514901
config385	0.859504174
config413	0.859464508
config414	0.859464508
config388	0.859444497
config389	0.859444497
config390	0.859444497
config461	0.859396362
config462	0.859396362
config130	0.858899594
config131	0.858899594
config132	0.858899594
config124	0.85868448
config125	0.85868448
config126	0.85868448
config235	0.834112129
config236	0.834112129
config237	0.834112129
config229	0.833604346
config230	0.833604346
config231	0.833532615
config223	0.832983075
config224	0.832983075
config225	0.832983075
config259	0.830969649
config260	0.830815687
config261	0.830815687
config253	0.8303838
config254	0.830229778
config255	0.830229778
config249	0.829753226
config247	0.829621968
config248	0.829621968
config217	0.828777216
config238	0.828631693
config239	0.828631693
config240	0.828631693
config218	0.828392076
config219	0.828392076
config283	0.828223234

config284	0.828223234
config285	0.828223234
config232	0.828123449
config233	0.828123449
config234	0.828123449
config226	0.828082719
config227	0.828082719
config228	0.828082719
config278	0.827464466
config277	0.827416762
config279	0.827416762
config256	0.826954721
config257	0.826954721
config258	0.826954721
config220	0.82685802
config221	0.82685802
config222	0.82685802
config271	0.826603035
config272	0.826603035
config273	0.826564934
config241	0.82557042
config242	0.82557042
config243	0.825553517
config280	0.82527
config281	0.82527
config282	0.82527
config523	0.82473023
config514	0.824725405
config524	0.824651449
config525	0.824651449
config515	0.824646589
config516	0.824646589
config250	0.824608434
config251	0.824608434
config252	0.824608434
config262	0.824535299
config263	0.824535299
config264	0.824535299
config517	0.824207117
config511	0.824128336
config519	0.824128336
config512	0.824049512
config518	0.823938372
config505	0.823918301
config513	0.823859519
config506	0.823531185

config507	0.823531185
config267	0.823351808
config526	0.823196076
config527	0.82311726
config528	0.82311726
config265	0.823107607
config266	0.823107607
config244	0.822940686
config245	0.822940686
config246	0.822940686
config508	0.822911637
config509	0.822911637
config510	0.822911637
config520	0.822617471
config521	0.822483279
config522	0.822483279
config286	0.822313166
config287	0.822313166
config288	0.822313166
config538	0.822049301
config539	0.821969937
config540	0.821969937
config547	0.821735106
config548	0.821735106
config549	0.821735106
config541	0.82152363
config543	0.8214443
config535	0.821253579
config542	0.821253579
config536	0.821174175
config537	0.821174175
config274	0.820583592
config275	0.820583592
config276	0.820583592
config550	0.820453033
config551	0.82043238
config552	0.82043238
config529	0.820292862
config530	0.820292862
config544	0.819929469
config545	0.819794625
config546	0.819794625
config268	0.819778638
config269	0.819778638
config270	0.819628486
config532	0.819513741

config533	0.819513741
config534	0.819082977
config531	0.818985484
config553	0.818324441
config554	0.818163977
config556	0.817721387
config557	0.817721387
config559	0.817172657
config561	0.817172657
config574	0.817158626
config575	0.817158626
config576	0.817158626
config562	0.817091462
config563	0.817091462
config564	0.817091462
config565	0.817084122
config568	0.816927149
config569	0.816927149
config570	0.816927149
config566	0.81680294
config560	0.816731115
config571	0.816664905
config573	0.816664905
config567	0.816642694
config572	0.81651752
config555	0.815921194
config558	0.815739104

10.7. Ejemplo análisis modelo de lenguaje

Tabla 10.6: Errores, contextos, sugerencias, distancia entre sugerencias y error, frecuencia de cada sugerencia en el contexto (obtenido por el servicio PhraseFinder) y corrección, del paso dos del ejemplo presentado en el capítulo de resultados.

Error	Contexto izq	Contexto der	Sugerencias	Dist.	Frecuencia	Corrección
qua	En lo		quia	1	0	que
			gua	1	0	
			qu	1	1226	
			qui	1	107	
			quo	1	1282	
			que	1	6285478	
			qué	1	1918	
rospectq	lo que	a la	respecta	2	2206642	respecta
			respecte	2	1058	
			respectá	2	0	
			respecté	2	0	

			respectó	2	0	
			respecto	2	64753	
			prospecta	2	0	
			prospecte	2	0	
			prospecto	2	0	
			prospectá	2	0	
			prospecté	2	0	
			prospectó	2	0	
nonramionto	y		honramiento	3	0	nombramiento
			nombramiento	3	90766	
doi	nombramiento	Jefe	dni	1	0	dio
			di	1	238	
			dio	1	358	
			dom	1	0	
			don	1	244	
			dos	1	284	
			doy	1	0	
rasÓn		por la	rascón	1	0	razón
			rasgón	1	0	
			raspón	1	0	
			masón	1	0	
			rosón	1	0	
			rabón	1	0	
			radón	1	0	
			ramón	1	359	
			ratón	1	349	
			rayón	1	85	
			razón	1	696738	
			rasan	1	0	
			rasen	1	0	
			rasó	1	0	
funcitmoM	las		fruncimos	3	0	funciono
			uncimos	3	0	
			fundimos	3	0	
			fungimos	3	0	
			funciono	3	146	
correapondienteo		como	correspondientes	2	3563	correspondiente
			correspondiente	2	6379	
coop			scoop	1	3176	cook
			chop	1	13418	
			copo	1	100610	
			cook	1	366294	
=-el			rel	1	65432	el
			tel	1	1177996	
			bel	1	290250	
			del	1	1921539862	
			el	1	3945850358	
			gel	1	211720	
asesoramientc	el	a los	asesoramiento	1	11328	asesoramiento
arcan			arcana	1	44514	marcan
			arican	1	704	
			marcan	1	864066	
			arcano	1	176822	
			aran	1	157162	
			ardan	1	19052	
			arfan	1	3298	
			arman	1	155082	
			arpan	1	298	

			arcén	1	10512	
			arcón	1	55062	
			arcar	1	1814	
			arcas	1	630146	
			arca	1	819224	
avíos-			avíos	1	59616	avíos
focha	hasta la		pocha	1	0	fecha
			rocha	1	0	
			tocha	1	0	
			ñocha	1	0	
			bocha	1	0	
			cocha	1	90	
			gocha	1	0	
			locha	1	0	
			mocha	1	0	
			facha	1	322	
			fecha	1	942663	
			ficha	1	164	
			foca	1	0	
do	tomado conocimiento	denuncias sobre	2do	1	0	de
			fdo	1	0	
			ado	1	0	
			dto	1	0	
			duo	1	0	
			rdo	1	0	
			vdo	1	0	
			dio	1	0	
			dom	1	0	
			don	1	0	
			dos	1	0	
			doy	1	0	
			dúo	1	0	
			ido	1	0	
			ldo	1	0	
			fo	1	0	
			jo	1	0	
			lo	1	0	
			no	1	0	
			o	1	0	
			ro	1	0	
			yo	1	0	
			ño	1	0	
			dr	1	0	
			du	1	0	
			db	1	0	
			dg	1	0	
			dl	1	0	
			dm	1	0	
			dá	1	0	
			dí	1	0	
			d	1	0	
			da	1	0	
			de	1	9410	
			di	1	0	
			dé	1	0	
irrcgulítridades	denuncias sobre	que se	irregularidades	3	1111	irregularidades
=-Decreto		del Poder	decreto	1	66541	decreto

es-	formuladas por		es	1	0	esa
			esa	1	224	
			ese	1	157	
			eso	1	0	
crltoy-—						
do	la Facultad	Medicina	2do	1	0	de
			fdo	1	0	
			ado	1	0	
			dto	1	0	
			duo	1	0	
			rdo	1	0	
			vdo	1	0	
			dio	1	76	
			dom	1	0	
			don	1	209	
			dos	1	135	
			doy	1	0	
			dúo	1	0	
			ido	1	0	
			ldo	1	0	
			fo	1	0	
			jo	1	0	
			lo	1	1519	
			no	1	8122	
			o	1	13849	
			ro	1	0	
			yo	1	0	
			ño	1	0	
			dr	1	102	
			du	1	0	
			db	1	89	
			dg	1	0	
			dl	1	0	
			dm	1	0	
			dá	1	0	
			dí	1	0	
			d	1	1218	
			da	1	2128	
			de	1	3786289	
			di	1	531	
			dé	1	989	
encomendándome		la	encomendando	3	11314	encomendando
raaliaaoidn	encomendando la	de una	realización	4	2197737	realización
			presuma	4	0	
			presumamos	4	0	
			presuman	4	0	
			presumas	4	0	
			presume	4	0	
			presumen	4	0	
			presumes	4	0	
			presumible	4	0	
			presumibles	4	0	
			presumid	4	0	
			presumida	4	0	
			presumiera	4	0	
			presumiere	4	0	
			presumiereis	4	0	

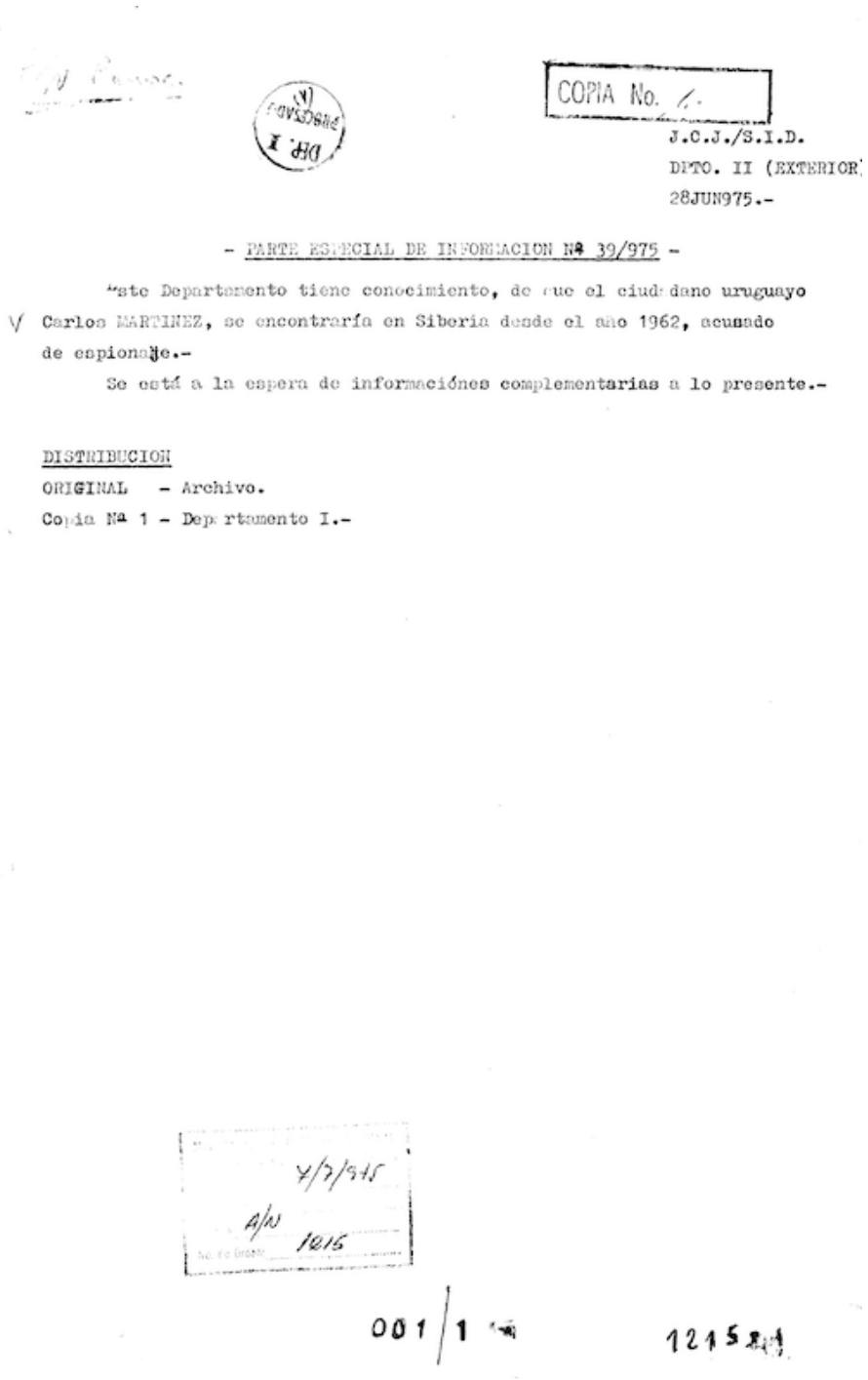
			presumieren	4	0	
			presumieres	4	0	
			presumiese	4	0	
			presumiesen	4	0	
			presumieses	4	0	
			presumirá	4	0	
			presumirán	4	0	
			presumirás	4	0	
			presumiré	4	0	
			presumiréis	4	0	
			presumiría	4	0	
			presumirían	4	0	
			presumirías	4	0	
			presumiste	4	0	
			presumió	4	0	
			presumáis	4	0	
			presumíais	4	0	
			presumido	4	0	
			presumir	4	0	
			rezumare	4	0	
			rezumareis	4	0	
			rezumaren	4	0	
			rezumares	4	0	
			resudare	4	0	
			resudareis	4	0	
			resudaren	4	0	
			resudares	4	0	
			perfumare	4	0	
			perfumareis	4	0	
			perfumaren	4	0	
			perfumares	4	0	
			presagie	4	0	
			presagien	4	0	
			presagies	4	0	
			prestare	4	0	
			prestareis	4	0	
			prestaren	4	0	
			prestares	4	0	
			prestarse	4	0	
			presidario	4	0	
			apresurare	4	0	
			apresurareis	4	0	
			apresuraren	4	0	
			apresurares	4	0	
			apresurarse	4	0	
			presurice	4	0	
			presuricen	4	0	
			presurices	4	0	
			presuricé	4	0	
			presuriza	4	0	
			presurizo	4	0	
			presurizá	4	0	
			presurizó	4	0	
			sumarie	4	0	
			sumarien	4	0	
			sumaries	4	0	
			sumarial	4	183	
oatoada			atoada	1	0	atoada

po			peo	1	46886	por
			pio	1	450962	
			pol	1	340294	
			tpo	1	54676	
			upo	1	92540	
			pon	1	487238	
			pop	1	381442	
			por	1	1492300748	
			pos	1	1593154	
			pro	1	7017544	
			pío	1	1590686	
			fo	1	513458	
			jo	1	1316598	
			op	1	6558220	
			lo	1	680926444	
			no	1	1103634742	
			o	1	518055922	
			ro	1	1819998	
			yo	1	80020802	
			ño	1	333036	
			p	1	53463836	
			pc	1	917334	
			pg	1	410136	
			pl	1	673542	
			pp	1	28471888	
			ph	1	1125314	
			pe	1	2104700	
			pi	1	965458	
funcionarla		de este	funcionara	1	0	funcionaría
			funcionaria	1	0	
			funcionaría	1	56	
rÓEibulR						
do		acuerdo a	2do	1	0	de
			fdo	1	0	
			ado	1	0	
			dto	1	0	
			duo	1	0	
			rdo	1	0	
			vdo	1	0	
			dio	1	0	
			dom	1	0	
			don	1	0	
			dos	1	0	
			doy	1	0	
			dúo	1	0	
			ido	1	0	
			ldo	1	0	
			fo	1	0	
			jo	1	0	
			lo	1	0	
			no	1	0	
			o	1	259	
			ro	1	0	
			yo	1	0	
			ño	1	0	
			dr	1	0	
			du	1	0	

			db	1	0	
			dg	1	0	
			dl	1	0	
			dm	1	0	
			dá	1	0	
			dí	1	0	
			d	1	129	
			da	1	864	
			de	1	2331146	
			di	1	136	
			dé	1	461	
exprosado	a lo		expresado	1	28635	expresado
so	solicito	me	aso	1	0	se
			soc	1	0	
			eso	1	0	
			oso	1	0	
			sao	1	0	
			seo	1	0	
			sol	1	0	
			son	1	0	
			sor	1	0	
			sos	1	0	
			soy	1	0	
			sto	1	0	
			uso	1	0	
			éso	1	0	
			fo	1	0	
			jo	1	0	
			lo	1	0	
			no	1	0	
			o	1	0	
			ro	1	0	
			yo	1	0	
			ño	1	0	
			s	1	0	
			sr	1	0	
			ss	1	0	
			sé	1	0	
			se	1	2439	
			si	1	0	
			su	1	0	
			sí	1	0	
pormita	se me	tomar conocimiento	dormita	1	0	permita
			permita	1	8528	
ce	que	me comunica	ace	1	0	se
			cae	1	65	
			cte	1	0	
			cíe	1	0	
			ice	1	0	
			cea	1	0	
			ces	1	0	
			che	1	0	
			je	1	2561	
			ne	1	71	
			be	1	99	
			de	1	268	
			e	1	47	
			fe	1	1063	

			ge	1	48	
			he	1	112	
			le	1	493	
			me	1	981	
			pe	1	0	
			re	1	44	
			se	1	810797	
			te	1	3364	
			ve	1	142	
			ye	1	47	
			cd	1	0	
			có	1	0	
			c	1	0	
			ca	1	0	
			ch	1	0	
			cl	1	0	
			cm	1	0	
esmunlco			esculco	2	0	esculco

10.6. Ejemplo documento Bleualign



COPIA No. 1

J.C.J./S.I.D.
DPTO. II (EXTERIOR)
28JUN975.-

- PARTE ESPECIAL DE INFORMACION Nº 39/975 -

Este Departamento tiene conocimiento, de que el ciudadano uruguayo Carlos MARTINEZ, se encontraría en Siberia desde el año 1962, acusado de espionaje.-

Se está a la espera de informaciónes complementarias a lo presente.-

DISTRIBUCION
ORIGINAL - Archivo.
Copia Nº 1 - Departamento I.-

4/2/975
4/N 1015

001/1

1215

Imagen 10.2: Documento completo de ejemplo 1 de la Subsección 6.1.1