PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

Generación automática de un corpus de comprensión lectora para el español a partir de un conjunto de datos en lengua inglesa

TESIS PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO INFORMÁTICO

AUTOR

Fabricio Andrés Monsalve Escudero

ASESOR:

Mag. Félix Arturo Oncevay Marcos

Lima, agosto, 2019

RESUMEN

Desde la aparición del computador, la comprensión lectora automática ha sido un tema de interés científico, resultando en diversas investigaciones y técnicas que le permitan a una máquina "comprender" diversos textos. La introducción del aprendizaje de máquina originó un gran cambio en este ámbito de estudio. Esto debido a que mientras los algoritmos de aprendizaje de máquina y procesamiento de lenguaje natural iban evolucionando, se necesitaba mayores cantidades de datos o ejemplos para poder aprender correctamente. Este problema fue abordado por varios estudios científicos, dando origen a un grupo significativo de conjuntos de datos enfocados a distintos tipos de comprensión lectora. Sin embargo, estos conjuntos de datos fueron creados solo para el idioma inglés ya que, hasta la actualidad, los trabajos relacionados a este ámbito se desarrollan en ese idioma. Por ello, hay pocas investigaciones enfocadas en comprensión lectora para otros idiomas como el español, ya que la creación de los conjuntos de datos necesarios demanda una gran cantidad de recursos (horas-hombre de expertos) para lograr un resultado de calidad, lo que hace muy costoso este objetivo.

Por lo tanto, se propone una solución de menor costo, apoyándonos en la traducción y validación automática de un conjunto de datos de inglés a español. Específicamente, el conjunto de datos *Stanford Question Answering Dataset* (SQuAD), desarrollado por la Universidad de Stanford para la tarea de comprensión de lectura en inglés, cuenta con más de 100,000 pares de preguntas-respuestas planteadas sobre múltiples artículos de Wikipedia, y donde la respuesta a cada pregunta es un segmento de texto contenido explícitamente en los párrafos del artículo. Para lograr este objetivo, se usarán modelos de traducción automática y métricas de validación automática para traducción, para consecuentemente poder entrenar un modelo algorítmico de comprensión lectora en español, el cual podría permitir alcanzar los resultados del estado del arte para el inglés.

Posteriormente, se desarrollará una interfaz de programación de aplicaciones (API), la cual servirá para la presentación de los resultados obtenidos.

Esta solución representa un desafío computacional e informático debido al gran volumen de datos a tratar, para lo cual se deben realizar procesos eficientes y una correcta utilización de recursos, manteniendo así la viabilidad del proyecto. Asimismo, el uso y aplicación de los resultados obtenidos en este proyecto es de gran variedad, ya que, a partir del entrenamiento de un modelo algorítmico de comprensión lectora, se puede aplicar en sistemas de extracción de información, sistemas de tutoría inteligente, preguntas frecuentes, entre otros. Además, este proyecto busca dejar un precedente y brindar un punto de partida en futuras investigaciones sobre generación automática de conjuntos de datos para comprensión lectora en español, utilizando un enfoque en aprendizaje de máquina y procesamiento de lenguaje natural.

A mis padres Patty y Nilton,

A mis abuelos y tíos,

A mi hermano Marcelo,

A todos mis amigos y su apoyo incondicional,

A mis profesores,

A todos los que aspiran a lograr un impacto en el mundo mediante la tecnología.

Tabla de Contenidos

Índice de Figuras

Índice de Tablas

Capítulo 1. Generalidades

1.1 Problemática

La comprensión lectora ha sido un tema de interés durante muchos años, una capacidad que se creía solo posible de un ser racional, enfocada en entender lo que se lee y haciendo referencia al significado de las palabras que conforman un texto (Pérez Zorilla, 2005).

Desde la aparición del computador los científicos han estado investigando y perfeccionando técnicas que le permitan a una máquina "comprender" textos, uno de los primeros intentos se remonta al año 1964 con el programa STUDENT, presentado en el MIT (Bobrow, 1964). Así, se logró mostrar cómo una computadora puede entender una entrada sencilla en lenguaje natural para solucionar problemas algebraicos de palabras. Luego se presentó ELIZA (Weizenbaum, 1976), el cual podía mantener una conversación coherente en inglés sobre cualquier tema. ELIZA obtuvo bastante popularidad como un proyecto juguete y se puede ver como un temprano precursor de los actuales sistemas comerciales enfocados a la interacción textual con usuarios, como los utilizados por *Ask.com*[1].

Hasta los años 80, la mayoría de sistemas que involucraban Procesamiento de Lenguaje Natural (NLP por sus siglas en inglés), que es el área enfocada en la comprensión del lenguaje humano por el computador, se basaban en un complejo conjunto de reglas diseñadas a mano (Liddy, 2001). Sin embargo, la introducción de los algoritmos de aprendizaje de máquina a finales de la década causó una gran diferencia. Debido a esto, se empezaron a utilizar métodos de aprendizaje de máquina (ML, por sus siglas en inglés) para clasificar texto, un ejemplo es el IBM Watson (Ferrucci et al., 2010). Sin embargo, según Searle (2011), Watson no comprendió

[1] https://www.ask.com

ni procesó en su totalidad las preguntas que se le dieron, por lo que no se lo clasificó como Comprensión de Lenguaje Natural en sí.

La evolución de las técnicas de ML y NLP requirieron el desarrollo de diversos conjuntos de datos, ya que los tipos de métodos requieren de grandes cantidades de datos (son *data hungry*), ejemplos, o experiencias previas para aprender (Bishop, 2006). En este contexto, se tiene como punto de quiebre el año 2015. Antes del 2015 existían 2 conjuntos de datos relativamente conocidos para la comprensión lectora: MCTest (Richardson et. al, 2013) con un total de 2600 preguntas y ProcessBank (Berant et. al, 2014) con 500 preguntas. Luego, a partir del año 2015 ya se tenían ocho nuevos conjuntos de datos (Danqi Chen, 2017):

- CNN/Daily Mail (Chen et al., 2016)
- Children Book Test – **Facebook** (Hill et al., 2015)
- Wiki Reading – **Google** (Hewlett et al., 2016)
- LAMBADA – **Cornell University** (Paperno et al., 2016)
- SQuAD 1.0 – **Stanford University** (Rajpurkar et al., 2016)
- Who did What – **Toyota Technological Institute at Chicago** (Onishi et al., 2016)
- NewsQA – **Maluuba** (Trischler et al., 2016)
- MS MARCO – **Microsoft** (Nguyen et al., 2016)

De la lista de conjuntos de datos anterior, todos han sido desarrollados para el idioma inglés, al igual que las investigaciones que los utilizan, dejando aún por explorar las posibilidades en el idioma español (Dascalu et al., 2017). Actualmente se conducen pocas investigaciones y proyectos relacionados a Comprensión de Lenguaje Natural, específicamente *question answering*, orientados al español, debido a que una de las principales barreras es la insuficiencia de los conjuntos de datos dedicados al entrenamiento de estos modelos (Dascalu et al., 2017).

En los últimos años, varios grupos se han beneficiado del desarrollo de estas investigaciones relacionadas a la comprensión de textos y *question answering* en el idioma inglés (Allen, Snow, & McNamara, 2015). Uno de los grupos que se ha beneficiado considerablemente es el de los Sistemas de Tutoría Inteligente (ITS por sus siglas en inglés), los cuales tienen como objetivo generar conocimiento en los usuarios sin la presencia de un tutor físico, y que aprovechan estos modelos de aprendizaje predictivo para mejorar la comprensión lectora en sus usuarios reforzando su técnica de autoaprendizaje (Dascalu et al., 2017). Otro grupo importante es el de estudiantes que buscan mejorar su nivel de comprensión lectora o tienen la necesidad consumir grandes cantidades de información proveniente de textos en un corto periodo de tiempo (Allen et al., 2015).

El problema principal que se busca solucionar en este proyecto de fin de carrera es la falta de un conjunto de datos dedicado al entrenamiento de modelos predictivos de comprensión lectora y *question answering* en español. Dado este problema, se propone una solución utilizando distintas técnicas como traducción automática y la implementación de una herramienta encargada de realizar la validación de la traducción resultante, garantizando la calidad de las traducciones, para poder entrenar modelos predictivos ya establecidos de comprensión de textos y *question answering*.

De esta manera, se generará un conjunto de datos y se implementará un modelo de aprendizaje predictivo que pueda realizar un proceso de comprensión lectora en español. Asimismo, se espera lograr una herramienta que facilite el análisis de documentos de texto, además de ser capaz de responder preguntas planteadas sobre el contenido del mismo en lenguaje natural y teniendo como hipótesis la utilización de las técnicas ya mencionadas para la obtención de un conjunto de datos mediante la traducción de uno en inglés, como es el caso de SQuAD (Rajpurkar et al., 2016), para poder aplicar los mismos métodos y procedimientos utilizados en las investigaciones para el idioma inglés, realizando un proceso de *bootstrapping*, el cual

consiste en construir un nuevo modelo algorítmico sobre uno ya existente mediante la traducción de su conjunto de datos de entrenamiento, lo cual ya ha sido realizado satisfactoriamente para otros idiomas (Gaspers, Karanasou, & Chatterjee, 2018). Para esto, se propone utilizar modelos de traducción automática y validación de la calidad de la traducción utilizando métodos como *backward translation*, la cual consiste en traducir nuevamente al idioma origen el texto traducido para comparar la calidad de la traducción (Gaspers et al., 2018). Posteriormente, utilizando técnicas de ML y NLP se entrenará un modelo predictivo de comprensión lectora, capaz de responder a preguntas en lenguaje natural sobre un texto, y finalmente se implementará una interfaz de programación de aplicaciones (API por sus siglas en inglés) la cual encapsulará los métodos de este modelo y le permitirá ser utilizado por aplicaciones de terceros, siendo el caso de esta investigación, pero no limitándose a, una interfaz web.

1.2 Objetivos:

En esta sección se indicarán los objetivos que se desea alcanzar durante el desarrollo del presente proyecto de fin de carrera.

1.2.1 Objetivo General:

Generar un conjunto de datos traducido del idioma inglés al español para el entrenamiento de un modelo algorítmico de comprensión lectora en español.

1.2.2 Objetivos Específicos:

- O1: Implementar un algoritmo que permita obtener un corpus de comprensión lectora en español tomando como base uno desarrollado nativamente para el idioma inglés.
- O2: Implementar un modelo algorítmico de comprensión de lectura en español para obtener la respuesta a una pregunta planteada, sobre un pasaje de texto, en lenguaje natural.
- O3: Implementar un componente de software para encapsular las funcionalidades del modelo generado y permitir su uso por terceros.

1.2.3 Resultados esperados:

- R1: Para el objetivo 1.- Componente de software para la lectura de datos y traducción automática de texto del idioma inglés al español.
- R2: Para el objetivo 1.- Componente de pre-procesamiento de texto y cálculo de métricas para la validación automática de la traducción de acuerdo a BLEU, ROUGE y similitud vectorial semántica.
- R3: Para el objetivo 1.- Experimentación numérica de las métricas calculadas sobre un proceso de *backtranslation* para el procesamiento del conjunto de datos.

- R4: Para el objetivo 2.- Modelo de línea base entrenado sobre el conjunto de datos inicial traducido, sin procesamiento.
- R5: Para el objetivo 2.- Modelo algorítmico entrenado sobre el conjunto de datos traducido y procesado.
- R6: Para el objetivo 3.- Interfaz de programación de aplicaciones (API).

1.2.4 Mapeo de objetivos y resultados esperados

Tabla 1: Mapeo de objetivos y resultados (elaboración propia)

Objetivo: Implementar un algoritmo que permita obtener un corpus de comprensión lectora en español tomando como base uno desarrollado nativamente para el idioma inglés.		
Resultado	**Meta física**	**Medio de verificación**
Componente de software para la lectura de datos y traducción automática de texto desde el idioma inglés al español.	- Software - Conjunto de datos traducidos sin procesar	- Revisión manual sobre una muestra aleatoria de los datos procesados
Componente de pre-procesamiento de texto y cálculo de métricas para la validación automática de la traducción de acuerdo a BLEU, ROUGE y similitud vectorial semántica.	- Software	- Pasaje de texto procesado. - Score (BLEU, ROUGE) obtenido por un pasaje de texto procesado.
Experimentación numérica de las métricas calculadas sobre un proceso de *backtranslation* para el	- Software - Conjunto de datos filtrado	- Valores de la distribución de valores de las métricas luego de aplicar la

procesamiento del conjunto de datos.		experimentación númerica.
Objetivo: Implementar un modelo algorítmico de comprensión de lectura en español para obtener la respuesta a una pregunta planteada, sobre un pasaje de texto, en lenguaje natural.		
Resultado	**Meta física**	**Medio de verificación**
Modelo de línea base entrenado sobre el conjunto de datos inicial traducido, sin procesamiento.	Modelo algorítmico	- Salida del modelo sobre un pasaje de texto y pregunta de prueba.
Modelo algorítmico entrenado sobre el conjunto de datos traducido y procesado.	Modelo algorítmico	- Salida del modelo sobre un pasaje de texto y pregunta de prueba.
Objetivo: Implementar un componente de software para encapsular las funcionalidades del modelo generado y permitir su uso por terceros.		
Resultado	**Meta Física**	**Medio de verificación**
Interfaz de programación de aplicaciones (API).	Software	- Interfaz web de usuario que consuma el API y permita interacción. - Consulta http al servicio generado

1.3 Herramientas y métodos

En esta sección se detallarán las distintas herramientas a utilizar en el desarrollo del presente proyecto de fin de carrera y la metodología a seguir para realizarlo de manera satisfactoria.

1.3.1 Herramientas

En la siguiente sección se presentarán y describirán las herramientas utilizadas en el presente trabajo de investigación.

Python

Python ("Python Software Foundation," 2001) es un lenguaje de programación orientado a objetos de fácil aprendizaje y uso de código abierto. Posee amplia cantidad de librerías con diferentes utilidades y objetivos, entre ellos aprendizaje de máquina. Actualmente, Python es uno de los lenguajes más utilizados en las áreas de aprendizaje de máquina debido a su capacidad para el procesamiento de datos y el conjunto de herramientas que posee. Por estas razones se ha elegido Python para el desarrollo del proyecto.

Jupyter Notebook

Jupyter Notebook ("Project Jupyter," 2015) es una aplicación de código abierto que permite compartir y crear documentos que contienen código (entre estos Python), ecuaciones y visualizaciones. Sus aplicaciones incluyen la limpieza de datos, análisis y transformación numérica, modelos estadísticos, aprendizaje de máquina y visualización de datos. En el presente proyecto, se empleará esta aplicación para exponer de manera ordenada el código y visualizar resultados.

Seaborn

Seaborn (Waskom, 2012) es una librería en Python para la visualización de datos y resultados. Provee interfaces de alto nivel para mostrar gráficos estadísticos. Esta librería será utilizada para mostrar los datos de manera visual y facilitar su exploración y preprocesamiento.

Scikit-learn

Scikit-learn (Pedregosa et al., 2012) o también conocido como *sklearn*, es un módulo integrado de Python con un amplio rango de algoritmos de aprendizaje de máquina para problemas de mediana escala utilizando métodos supervisados y no supervisados.

SpaCy

SpaCy (Honnibal, 2015) es una librería de código abierto para procesamiento de texto y Procesamiento de Lenguaje Natural avanzado en Python y Cython.

Posee el *syntactic parser* más rápido del mundo. Actualmente ofrece modelos estadísticos de redes neuronales para inglés, alemán, español, portugués, francés, italiano, holandés y *Named-entity Recognition****(NER)*** multilingüe, así como tokenización para varios otros idiomas. En el presente proyecto se utilizará esta herramienta para el análisis y pre procesamiento de la data.

Gensim

Gensim (Řehůřek & Sojka, 2010) es una librería creada para realizar modelamiento semántico no supervisado de texto plano.

Sus usos más importantes son modelamiento de tópicos, indexación de documentos y *similarity retrieval* para conjuntos de datos grandes. Posee la ventaja de que sus algoritmos son *memory-independent*, esto quiere decir que puede procesar entradas más grandes que la RAM.

Pandas

Pandas (Mckinney, 2011) es una librería escrita en Python para la manipulación de data y análisis. En particular ofrece estructuras de datos y operaciones para manipular tablas numéricas y series temporales. Esta librería posee herramientas muy útiles para leer y escribir datos entre las estructuras de datos en memoria y diferentes formatos de archivos. Se utilizará esta herramienta para la manipulación de los conjuntos de datos tanto de entrenamiento y prueba.

Natural Language Toolkit

Natural Language Toolkit (Bird, Klein, & Loper, 2009) o más comúnmente NLTK, es un conjuntos de bibliotecas y programas para el procesamiento simbólico y estadístico del lenguaje natural (PLN) para inglés en el lenguaje de programación Python. Es compatible con funcionalidades de clasificación, tokenización, derivación, etiquetado, análisis sintáctico y razonamiento semántico. Esta herramienta será utilizada en el proyecto para el pre procesamiento de los datos en inglés.

PyTorch

PyTorch (Paszke et al., 2017) es una librería de *Machine Learning* de código abierto para Python y es utilizada para aplicaciones como Procesamiento de Lenguaje Natural (PLN). Es principalmente desarrollada por el grupo de investigación de inteligencia artificial de Facebook. Provee dos características de alto nivel como: cómputo de Tensor (como numpy) con significante aceleración GPU y Redes Neuronales Profundas construidas en un sistema *tape-based* de auto diferenciación. Esta herramienta será utilizada como reemplazo para el uso de algunas librerías como numpy y aprovechar el poder de los GPUs para obtener una mejor performance.

Google Cloud Translation API

Google cloud translation API es la interfaz de programación de aplicaciones desarrollada por Google la cual permite la traducción de una cadena de texto arbitraria hacia cualquier lenguaje soportado utilizando *state-of-the-art Neural Machine Translation*. Esta API es altamente receptiva, de esta manera permitiendo a sitios web y aplicaciones una integración rápida para una traducción eficiente y dinámica del texto fuente desde el idioma origen hacia el idioma objetivo. Esta herramienta tiene funcionalidades muy importantes y útiles como la detección automática del lenguaje, soporte de múltiples lenguajes, integración simple y altamente escalable. Esta herramienta al igual que otras de similar naturaleza, serán utilizadas en esta investigación para realizar el proceso de traducción automática.

Fast Text

Fast Text (Bojanowski, Grave, Joulin, & Mikolov, 2016) es una librería desarrollada por Facebook enfocada al aprendizaje eficiente de *word representations* y clasificación de oraciones. Incluye vectores para 157 lenguajes entrenados en Wikipedia y Crawl, y posee modelos para identificación de lenguaje y también modelos supervisados. Esta herramienta será utilizada en el proyecto para brindar apoyo al proceso de word embeddings.

Flask

Flask (Ronacher, 2010) es un *microframework* escrito en Python que permite crear aplicaciones web de manera rápida y poco código. Soporta extensiones que pueden añadir características de aplicación como si hubieran sido implementadas en Flask. Se utilizará esta herramienta para generar los servicios REST que conformarán la API.

1.3.2 Métodos y procesos

En la ilustración 1 se muestra un gráfico que representa la visión general del presente proyecto.

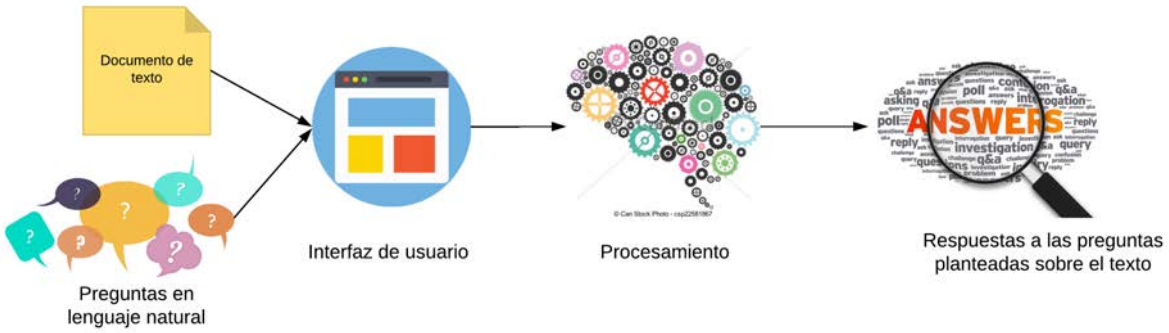


Ilustración 1: Flujo básico del proyecto (elaboración propia)

KDD

Knowledge Discovery in Databases (KDD) o descubrimiento de conocimiento en bases de datos, propuesto por Fayyad et al., es el proceso de extracción que consiste en 5 pasos iterativos entre sí. El proceso KDD empieza por determinar un objetivos y termina con la implementación del conocimiento descubierto (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). Los pasos utilizados son los siguientes:

1. *Seleccionar los datos significativos:* Escoger únicamente información relevante.
2. *Pre procesamiento:* En esta etapa, una vez con los datos obtenidos, se deben limpiar los datos, ya sean ruidosos o faltantes.
3. *Transformación de datos:* En esta etapa se generan mejores datos para la minería de datos. Esto incluye reducción de dimensiones, transformación de atributos y datos.
4. *Minería de datos:* Esta etaba se basa en la busqueda y descubrimiento de patrones de interés en los datos, aplicando tareas de descubrimiento como clasificación, patrones secuenciales, asociaciones, entre otros.

5. *Interpretación:* Una vez tenidos los resultados, es necesario representar los datos de forma apropiada para que sean examinados.

Metodología de desarrollo

En la ilustración 2 se presenta el proceso de ejecución del presente proyecto, mostrando los pasos se seguirán durante el desarrollo, relacionándolos con los objetivos y resultados

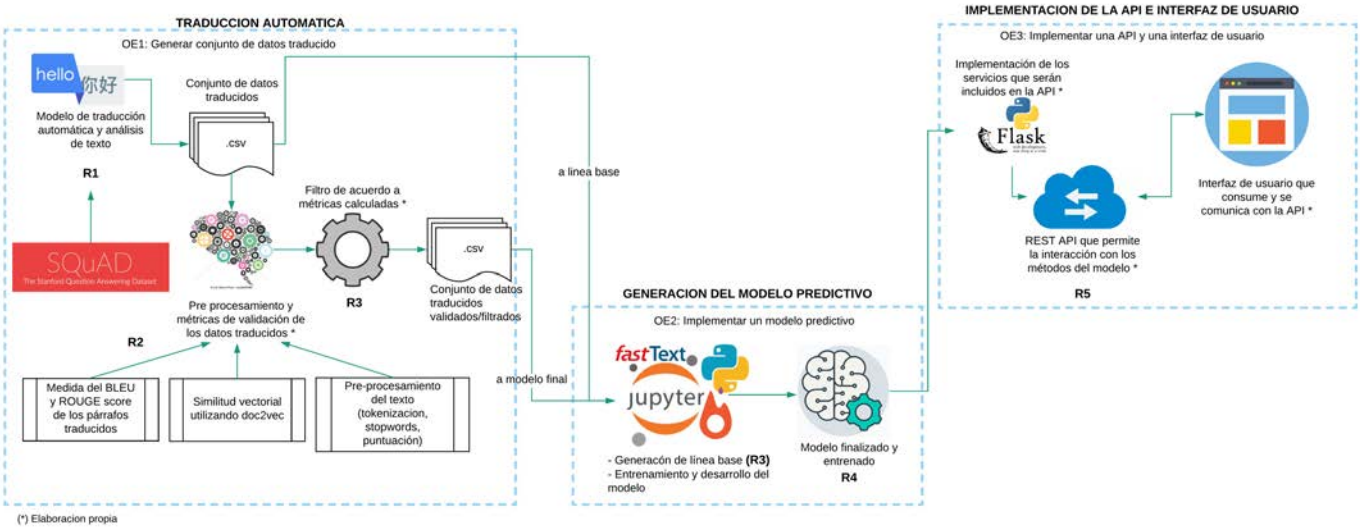


Ilustración 2: Proceso de desarrollo a seguir (elaboración propia)

especificados anteriormente.

- **Traducción automática**

  Las técnicas de traducción automática son utilizadas para generar el corpus en español, el cual será generado en base al conjunto de datos ya existente SQuAD (Rajpurkar et al., 2016), necesario para la implementación del modelo predictivo de comprensión de textos.

  Esta se realizará mediante un modelo algorítmico, por lo cual se utilizará la herramienta Google Cloud Translation API (Google LLC, n.d.) la cual está basada en *Neural Machine Translation*, con lo que se obtendrá una salida de texto traducido al idioma objetivo.

  Se generarán puntuaciones para la traducción de acuerdo a la calidad con la que se tradujo la palabra, frase u oración (Gaspers et al., 2018).

- **Validación de la traducción**

Se pueden utilizar varios métodos para calificar la calidad de la traducción y validar el resultado, en el presente proyecto se utilizarán un método principal con una variación: este será *backward translation* o *round-trip translation,* el cual consiste en traducir nuevamente al idioma original el resultado de una traducción para así comparar la integridad y coherencia del texto traducido (Gaspers et al., 2018). Sin embargo, por más que parezca un método suficiente para evaluar la traducción, se considera este método "pobre" para utilizarse únicamente (Somers, 2005) ya que no se está probando un solo sistema, sino dos: el par generado para traducir al lenguaje objetivo y el par generado de traducir de vuelta al lenguaje original.

La validación del método de *backward translation* se realizará de acuerdo a distintas métricas de validación automática para traducción, como BLEU (Papineni, Roukos, Ward, & Zhu, 2002) y ROUGE (Lin, 2004), los cuales se encargan de medir la ocurrencia de n-gramas del texto de original en el texto resultado de la traducción inversa y viceversa. A esto se le agregará la similitud entre los vectores de párrafos generados tanto para el texto original como para el texto traducido de vuelta, calculada mediante la distancia coseno y finalmente se hará una comparación entre tres categorías gramaticales importantes (verbos, adjetivos y sustantivos) para cada texto, midiendo que tan bien se preservaron estos valores.

Se utilizarán las herramienta NLTK (Bird et al., 2009) para el cálculo de la métrica BLEU, Gensim (Řehůřek & Sojka, 2010) para el cálculo de los vectores de párrafo y spaCy (Honnibal, 2015) para el etiquetado de categorías gramaticales.

- **Aplicación del modelo predictivo**

El modelo predictivo en español será implementado tomando como base modelos para el inglés que utilicen, en su mayoría, clasificadores y reconocimiento de entidades nombradas.

Lo que se busca es realizar un *bootstrapping* de un modelo en inglés a un nuevo idioma, en este caso, el español mediante la traducción de los conjuntos de entrenamiento.

En este proceso se utilizará el modelo de doc2vec (Le & Mikolov, 2014) con la herramienta Gensim (Řehůřek & Sojka, 2010) para hacer la representación vectorial de los párrafos, llamadas *"document vectors"*. Este modelo posee dos arquitecturas: *Distributed Memory version of Paragraph Vector* (PV-DM) y *Distributed Dag of Words version of Paragraph Vector (PV-DBOW)*, en este caso se utilizará PV-DBOW dado que calcula los vectores de las palabras del documento tomando en cuenta su contexto para luego infierir el vector del documento. El entrenamiento de los vectores se realizará sobre un *dump* de wikipedia, el cual contiene todos los artículos en inglés publicados a la fecha.

Como conjunto de entrenamiento será utilizado el corpus generado gracias al modelo de traducción automática menos una porción aleatoria que será utilizada como conjunto de prueba. La partición será generada con la herramienta Scikit-learn (Pedregosa et al., 2012).

- **Implementación de la API e interfaz de usuario**

Para esta parte se utilizará la herramienta Flask (Ronacher, 2010), con la cual se encapsularán los métodos del modelo predictivo entrenado anteriormente en servicios REST, los cuales podrán ser consumidos por un aplicativo web o cualquier aplicativo que soporte este tipo de servicios.

Los servicios serán ejecutados en un servidor, el cual será una computadora con conexión a internet y capacidad para ejecutar los métodos del modelo predictivo, de esta manera permitiendo la utilización y prueba del resultado de esta investigación desde casi múltiples dispositivos electrónicos con acceso a internet.

Paralelo a la implementación de la API, se implementará una interfaz de usuario, la cual será un aplicativo web simple que permita la comunicación bidireccional entre el usuario

que la utilice y los servicios que ofrece la API, en este caso aceptando una entrada de texto y las preguntas correspondientes, y devolviendo como respuesta el resultado generado por el modelo predictivo en el servidor.

- **Cuadro de herramientas y métodos**

Tabla 2 Cuadro de objetivos y herramientas a usar (elaboración propia)

Objetivo: Implementar un algoritmo que permita obtener un corpus de comprensión lectora en español tomando como base uno desarrollado nativamente para el idioma inglés.		
Resultado	**Medio de verificación**	**Herramientas y métodos**
Componente de software para la lectura de datos y traducción automática de texto desde el idioma inglés al español.	- Revisión manual sobre una muestra aleatoria de los datos procesados	- Google Cloud Translation API - Python - Jupyter - Pandas - Traducción automática
Componente de pre-procesamiento de texto y cálculo de métricas para la validación automática de la traducción de acuerdo a BLEU, ROUGE y similitud vectorial semántica.	- Pasaje de texto procesado. - Score (BLEU, ROUGE) obtenido por un pasaje de texto procesado.	- Python - Jupyter - Validación de la traducción - Numpy - Nltk - Gensim - Spacy - Pandas
Experimentación numérica de las métricas calculadas sobre un proceso de *backtranslation* para el	- Valores de la distribución de valores de las métricas luego de aplicar la	- Python - Jupyter - Pandas - Nltk

procesamiento del conjunto de datos.	experimentación númerica.	- Spacy - Gensim
Objetivo: Implementar un modelo algorítmico de comprensión de lectura en español para obtener la respuesta a una pregunta planteada, sobre un pasaje de texto, en lenguaje natural.		
Resultado	**Medio de verificación**	**Herramientas y métodos**
Modelo de línea base entrenado sobre el conjunto de datos inicial traducido, sin procesamiento.	- Salida del modelo sobre un pasaje de texto y pregunta de prueba.	- Jupyter - Scikit-learn - Numpy - PyTorch - Generación del modelo predictivo
Modelo algorítmico final entrenado sobre el conjunto de datos traducido y procesado.	- Salida del modelo sobre un pasaje de texto y pregunta de prueba.	- Jupyter - PyTorch - Numpy - Scikit-learn - Seaborn - Generación del modelo predictivo
Objetivo: Implementar un componente de software para encapsular las funcionalidades del modelo generado y permitir su uso por terceros.		
Resultado	**Medio de verificación**	**Herramientas y métodos**
Interfaz de programación de aplicaciones (API).	- Interfaz web de usuario que consuma el API y permita interacción. - Consulta http al servicio generado	- Python - Flask - Interfaz de web de usuario

1.4 Alcance y limitaciones

1.4.1 Alcance

El presente proyecto de fin de carrera se llevará a cabo partiendo de la selección de un conjunto de datos en el idioma inglés para ser traducido al español de manera automática.

Para el proceso de traducción automática se utilizará Google Cloud Translation API (Google LLC, n.d.), primero se traducirán todos los datos, validando la traducción con métodos como *backward translation*, para luego escoger las buenas traducciones de todas las traducciones obtenidas, i.e. las que se espera mejoren el rendimiento del modelo de comprensión lectora. Esta validación se realizará traduciendo el resultado obtenido por la herramienta al idioma original nuevamente, y de esta manera, se comparará la calidad de la traducción con el enunciado original. Para ello se utilizarán métricas de validación automática como BLEU (Papineni et al., 2002) y ROUGE (Lin, 2004), entre otros tipos de comparación cuantitativa como la comparación por similitud vectorial utilizando vectores de documentos o *paragraph vectors*.

Luego de este proceso, se generará una línea base con el conjunto de datos traducido sin la parte de post procesamiento ni validación de la traducción para, de esta manera, tener un punto de partida para medir la variación en precisión generada con el modelo predictivo a implementar. Una vez ya obtenido el corpus traducido y procesado, se procederá a implementar y adaptar el modelo algorítmico de comprensión de textos en base a modelos ya existentes para el inglés. Lo que se busca realizar es un proceso de *bootstrapping* de un modelo funcional para el idioma inglés al español.

Para finalizar, se implementará una Interfaz de Programación de Aplicaciones (API) la cual contendrá el modelo entrenado y será desarrollada en el lenguaje Python. Se desarrollará una

interfaz web de usuario, la cual consumirá la API y permitirá que cualquier usuario pueda utilizar las funcionalidades de este proyecto.

1.4.2 Limitaciones

El proceso de validación de la traducción será realizado de manera automática debido a que, al poseer un volumen de datos muy grande (aproximadamente 100,000 pares de preguntas y respuestas sobre más de 500 artículos), tener un profesional validando la traducción manualmente representaría un costo muy elevado.

Debido a que en el alcance no está contemplado el proponer un nuevo modelo de comprensión lectora, el proyecto se ve sujeto a resultados que alcanzan el estado del arte en el idioma inglés.

1.5 Viabilidad

1.5.1 Viabilidad técnica

Se posee libre acceso a los lenguajes de programación y herramientas a utilizar durante el desarrollo del proyecto. También quien suscribe posee experiencia utilizando el lenguaje Python y algunas herramientas mencionadas, además de contar con el recurso tecnológico (computadora) necesario para conducir el desarrollo del proyecto sin inconvenientes técnicos.

1.5.2 Viabilidad temporal

Para el desarrollo del presente proyecto, se estima una duración de 4 meses. En el Anexo 1 se presenta el cronograma de planificación de las tareas y tiempo planificado para su ejecución, observándose que es posible ejecutarlas en el tiempo adecuado.

1.5.3 Viabilidad económica

Debido a que las herramientas y datos a utilizar durante el desarrollo del proyecto son de código abierto y libre acceso, se evitan limitaciones financieras y de accesibilidad durante el desarrollo del proyecto.

1.6 Riesgos

En la tabla 3 se presentan los riesgos identificados en el proyecto y las medidas de contingencia y mitigación planificadas en caso sucedan.

Tabla 3: Tabla de riesgos del proyecto. [Elaboración propia]

Descripción	Síntomas	Probabilidad	Impacto	Severidad	Mitigación	Contingencia
No lograr traducir el conjunto de datos debido al costo que implica la traducción automática.	La API de traducción de Google cobra un monto de $20 por millón de caracteres.	0.5	0.9	0.8	Utilizar los créditos de prueba que otorga Google para realizar el proceso de traducción.	Entrenar un modelo de traducción automática propio.
No lograr obtener los resultados óptimos para el modelo predictivo debido al tiempo de procesamiento requerido.	Los modelos predictivos que requieren alta capacidad de procesamiento, usualmente GPU, demoran en entrenarse.	0.1	0.8	0.5	Utilizar un servidor dedicado, como los de Google, el Grupo de investigación IA PUCP o créditos educativos de Amazon AWS, que poseen GPUs potentes.	Reducir los parámetros del modelo y cantidad de datos a procesar.

Capítulo 2. Marco conceptual

A continuación, se hace una breve descripción de los conceptos teóricos que ayudarán a esclarecer algunos pasajes de la problemática. Se mencionan definiciones sobre ITS, Algoritmos, Aprendizaje de Máquina y Procesamiento de Lenguaje Natural.

2.1 Objetivos del Marco Conceptual

En el presente marco conceptual se busca definir los conocimientos necesarios para poder comprender la problemática de este proyecto para la generación de un corpus en español enfocado en entrenar un modelo predictivo para comprender texto y responder preguntas.

2.2 Conceptos de Ciencias de la Computación

2.2.1 Aprendizaje de Máquina (ML por sus siglas en inglés)

Es una rama de IA que puede ser definida como una serie de métodos computacionales que utilizan la experiencia para mejorar su rendimiento o para hacer predicciones más certeras. Por *experiencia* se refiere a la información pasada disponible, la cual toma forma de datos electrónicos recolectados y que son hechos disponibles para su análisis (Mohri, Rostamizadeh, & Talwalkar, 2012a).

Por ello se puede decir que ML consiste en el diseño preciso y eficiente de algoritmos de predicción, y dado que el éxito de estos depende de los datos que se utilicen, ML está relacionado con el análisis de datos y estadística (Mohri, Rostamizadeh, & Talwalkar, 2012b).

2.2.2 Procesamiento de Lenguaje Natural (NLP por sus siglas en inglés)

NLP es un campo de las ciencias de la computación, inteligencia artificial y lingüística que estudia las interacciones entre las computadoras y el lenguaje humano.

Consiste en la utilización de un lenguaje natural para comunicarnos con la computadora, debiendo ésta entender las oraciones que le sean proporcionadas, el uso de estos lenguajes naturales, facilita el desarrollo de programas que realicen tareas relacionadas con el lenguaje o bien, desarrollar modelos que ayuden a comprender los mecanismos humanos relacionados con el lenguaje (Chowdhury, 2005).

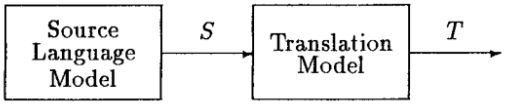
2.2.3 Comprensión del Lenguaje Natural (NLU por sus siglas en inglés)

Es una parte del procesamiento de lenguaje natural (NLP) en inteligencia artificial (AI) que trabaja con la comprensión de lectura para poder entender en un alto nivel el lenguaje humano en sus diferentes componentes, pero principalmente se enfoca en el semántico (sub-rama de la lingüística abocada a los significados de las palabras). Se le considera un problema *AI-Hard* (Yampolskiy, n.d.).

2.2.4 Traducción Automática (Machine Translation, MT)

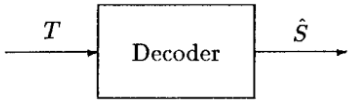
Es un sub-campo de la lingüística computacional que investiga el uso del software para traducir texto o habla de un lenguaje a otro.

El trabajo de un traductor es el de trasladar a un idioma el significado expresado por un pasaje de texto en otro idioma (Brown et al., n.d.). Si bien ningún sistema proporciona una traducción automática en alta calidad y totalmente automática del texto no restringido, muchos sistemas totalmente automatizados producen resultados razonables (Bar-Hillel, 1960).



$$\Pr(S) \times \Pr(T \mid S) = \Pr(S,T)$$

A *Source Language Model* and a *Translation Model* furnish a probability distribution over source-target sentence pairs (S,T). The joint probability $\Pr(S,T)$ of the pair (S,T) is the product of the probability $\Pr(S)$ computed by the language model and the conditional probability $\Pr(T \mid S)$ computed by the translation model. The parameters of these models are estimated automatically from a large database of source-target sentence pairs using a statistical algorithm which optimizes, in an appropriate sense, the fit between the models and the data.



$$\hat{S} = \underset{S}{\operatorname{argmax}} \Pr(S \mid T) = \underset{S}{\operatorname{argmax}} \Pr(S,T)$$

A *Decoder* performs the actual translation. Given a sentence T in the target language, the decoder chooses a viable translation by selecting that sentence $\hat{S}$ in the source language for which the probability $\Pr(S \mid T)$ is maximum.

Ilustración 3: Estructura de un sistema de Machine Translation (Brown et al., n.d.)

2.2.5 Machine Comprehension (también llamado text compehension, machine reading)

Se entiende por machine comprehension al proceso que realiza una máquina de proporcionar una respuesta textual respecto a una pregunta relacionada con un texto específico. Esta respuesta no debe contener información irrelevante a la pregunta y debe estar en calidad de ser comparada con una respuesta proporcionada por una persona, permitiendo a los hablante aceptar ambas respuestas (Chen et al., 2017).

Ilustración 4: Proceso de machine comprehension (Chen et al., 2017).

2.3 Conceptos Generales

2.3.1 Sistema de Tutoría Inteligente (ITS por sus siglas en inglés)

Es un sistema informático que tiene como objetivo proporcionar instrucción o retroalimentación inmediata y personalizada a los estudiantes (Psotka, Massey, & Mutter, 1988). Por lo general, funciona sin la intervención de un profesor humano.

Un ITS usualmente pretende replicar los beneficios demostrados de la tutoría uno-a-uno o personalizada, en contextos donde los estudiantes, de otra manera, solo tendrían acceso a una instrucción de uno-a-muchos por un solo maestro (por ejemplo, clases en el aula), o ningún maestro en absoluto (por ejemplo, tareas en línea). Un tutor inteligente, por lo tanto: "es un sistema de software que utiliza técnicas de inteligencia artificial (IA) para representar el conocimiento e interactúa con los estudiantes para ensenárselo" (Vanlehn, 2011).

2.3.2 Question Answering (QA)

Es un campo de IR y NLP que se basa en la construcción sistemas que respondan preguntas automáticamente. Ante una pregunta del usuario, mediante técnicas de IR se extraen pasajes directamente de los documentos, guiados por el texto de la pregunta (Dan Jurafsky & Martin, 2017). Un sistema de question answering provee respuestas directas a preguntas realizadas por los usuarios consultando su base de información. Sin embargo, la dificultad del procesamiento de lenguaje natural (NLP) ha limitado el alcance de los sistemas de question answering a los sistemas expertos específicos del dominio (Kwok, Etzioni, & Weld, 2001a).

Un sistema automatizado de QA basado en una colección de documentos, normalmente cuenta con 3 componentes principales (Kwok, Etzioni, & Weld, 2001b):

- Motor de IR, que maneja las solicitudes de recuperación

- Mecanismo de formulación de consultas, que traduce las preguntas en lenguaje natural a consultas que utilizará el motor de IR
- Extracción de respuesta, que analiza los documentos y extrae las respuestas

2.3.3 Bootstrapping

La frase *bootstrapping process* posee múltiples significados en distintas áreas de estudio, en el caso de este proyecto, esta frase es utilizada para hacer énfasis que el proceso (en nuestro caso el modelo algorítmico) es construido sobre sí mismo. La información obtenida durante la ejecución del proceso sirve para proporcionar el fondo o base necesarios para poder crear escenarios que se utilizarán para observar el rendimiento del estado objetivo en situaciones simuladas (Schraagen, Chipman, & Shalin, 2000).

En otras palabras, es la técnica de empezar con recursos existentes (como base) para crear, a partir de ellos, algo más complejo. En el caso de este proyecto esta definición se acopla a la utilización de un modelo algorítmico de comprensión lectora entrenado en inglés para la generación de un modelo en español.

Capítulo 3. Estado del Arte

En esta sección se tiene como objetivo verificar y mostrar empírica y teóricamente las limitaciones, conocimientos previos, barreras y problemas presentados en la generación de corpus para el entrenamiento de un modelo predictivo de comprensión de textos, tanto en inglés como en español.

3.1 Estrategia de búsqueda y discusión de fuentes

3.1.1 Definición de términos de búsqueda

Para la definición de estos, se tomó en cuenta lo siguiente:

- Búsquedas preliminares, utilizando distintas combinaciones de términos relacionadas al tema de investigación.
- Sinónimos y abreviaturas, si aplica.
- Revisión de resúmenes, títulos y palabras clave.
- Utilización de operadores lógicos AND y OR.

3.1.2 Selección de Fuentes

Se seleccionaron fuentes de búsqueda de acuerdo a la lista recomendada por la página web de la Pontificia Universidad Católica del Perú, revisiones sistemáticas en el área de ciencias de la computación. Siendo las bases de datos consultadas las siguientes:

- Scopus (http://www.scopus.com)
- Web of science (http://apps.webofknowledge.com/)
- Google scholar (http://scholar.google.com), el cuál se utilizó solo de manera complementaria.

3.1.3 Palabras clave

Se listarán las palabras clave de acuerdo a su relación entre ellas:

- Machine comprehension, machine reading, text comprehension
- Intelligent tutoring systems
- Question answering

3.1.4 Cadenas de búsqueda

De acuerdo a los resultados obtenidos en las búsquedas preliminares, se generaron más términos y combinaciones, resultando finalmente en tres cadenas de búsqueda con resultados significativos:

Tabla 4: Lista de cadenas de búsquedas y sus resultados

Cadena de búsqueda		**Resultados**
1	(text AND comprehension OR machine AND reading) AND spanish) AND (LIMIT-TO (SUBJAREA, "COMP"))	18
2	("intelligent tutoring system" AND (text AND comprehension OR machine AND reading)) AND (LIMIT-TO (SUBJAREA, "COMP"))	17
3	("text comprehension" OR "machine reading") AND "question answering" AND ((dataset OR "data set") AND generation) AND (LIMIT-TO (SUBJAREA,	22

La cadena número 1 se encuentra enfocada a encontrar investigaciones de comprensión de textos aplicadas al idioma español. La cadena número 2 se centra en buscar los distintos métodos utilizados en las investigaciones, aplicados a la comprensión de texto. Finalmente, la cadena número 3 busca encontrar investigaciones sobre la generación y creación de un conjunto de datos enfocados a la comprensión de textos y QA.

Cabe resaltar que la aplicación de la cadena de búsqueda se realizó utilizando el parámetro de ALL FIELDS, lo cual aumenta la cantidad de resultados obtenidos, salvo el caso de la cadena 2 que se limitó al título, resumen y palabras clave.

3.1.5 Resultados de búsqueda

De aquellos estudios, muchos se centran en investigar causas de la baja comprensión lectora y su repercusión en el desarrollo de sistemas que apoyen a mejorar este aspecto.

Por ello se seleccionaron los que se enfocaban mejor en resolver la problemática de esta investigación, primando el idioma español, generación de corpus y algoritmos de ML enfocados a resolver estos problemas.

Dado estos criterios, se seleccionaron 4 (que eran los que mejor se adaptaban), con los cuales se buscará responder las siguientes preguntas:

- ¿Sobre qué idioma se enfocó la investigación?
- ¿Cuáles fueron los métodos utilizados en el desarrollo de la investigación?
- ¿Cómo se generó el conjunto de datos utilizado?

3.2 Revisión y Discusión

En esta sección se presenta la revisión de investigaciones recientes desde diferentes enfoques científicos.

Teaching iSTART to Understand Spanish (Dascalu et al., 2017)

Consulta: 15/04/2019

Distintas técnicas de NLP han sido utilizadas para analizar las respuestas escritas de los estudiantes como un medio para medir su desempeño a través de múltiples ámbitos, como capacidad de redacción y comprensión de lectura. Sin embargo, los algoritmos aplicados están

limitados en su habilidad de generalizarse a varios lenguajes. En la investigación se habla sobre la herramienta iSTART (Interactive Strategy Training for Active Reading and Thinking), la cual se enfoca en proveer a estudiantes instrucción y práctica en estrategias de comprensión lectora, centrandose en la estragecia de auto explicación. Las respuestas de los estudiantes (*autoexplicaciones*) son puntuadas por un algoritmo dependiendo de cuan bien los estudiantes están utilizando la técnica de comprensión, el cual está basado en la combinación de medidas basadas en palabras y Análisis de la Semántica Latente (LSA por sus siglas en inglés).

El objetivo principal de la investigación yace en trasladar la herramienta iSTART al idioma español desde el inglés. Se presentaron múltiples complicaciones, desde cómo traducir los materiales de instrucción para que sean entendibles por la mayor parte de hablantes nativos de español hasta la selección de textos de práctica que podrían poseer un contexto significativo para muchos hispanohablantes.

El conjunto de datos se generó traduciendo de manera manual el conjunto de datos original en inglés.

Se propuso utilizar un algoritmo evolutivo basado en meta-heurísticas inspirado en genética, específicamente en procesos bio-inspirados (mutación, selección, cruce) para mejorar la precisión del algoritmo de calificación de la nueva herramienta (iSTART-E: iSTART en español). El propósito principal de este algoritmo fue el de determinar los coeficientes óptimos de la Función de Análisis Discriminante (DFA en inglés) que crean el mejor mapeo entre los resultados obtenidos en español con los que ya se tienen en inglés.

En la ilustración 5 se muestran los problemas que se presentaron durante el desarrollo de la investigación y la solución que se implementó.

Challenge	Implemented solution
Providing a new dictionary of words	Upon manual review, we selected the dictionary found at http://www.winedt.org/dict.html which includes low-frequency, scientific words
Introducing a new list of stop words	The stop words list from Snowball (http://snowball.tartarus.org/algorithms/spanish/stop.txt) was expanded to include words describing numbers and interjections (e.g., "bah")
Correcting misspelled words	Instead of the Soundex algorithm available for English, we implemented a rule matching algorithm that relies on the Levenshtein edit distance
Tagging important words from practice texts	The Stanford Core NLP for Spanish (http://stanfordnlp.github.io/CoreNLP) was integrated and used to identify content words (i.e., nouns, verbs, adjectives, adverbs)
Normalizing words to allow for comparisons	Our static lemmatizer based on predefined transformation http://www.lexiconista.com/datasets/lemmatization/ automatically changes word forms to their corresponding inflectional form, i.e. lemma
Building an LSA space in Spanish	The LSA space was built using Apache Mahout with the lemmatized corpus provided by El Grupo de Interés en el Análisis de la Semántica Latente (http://elsemantico.es/index.html)
Translating regular expressions	The algorithm uses regular expressions to identify various types of self-explanations and to flag special types of responses. Manual corrections were made due to language specificities
Iterative testing	We manually translated 2,982 English self-explanations from the identification mini-games into Spanish. Distributions of the various components captured by the algorithms (e.g., matching content words; LSA cosine values) and repeated measures ANOVAs for each component across the two languages helped us identify discrepancies

Ilustración 5: Desafíos encontrados y soluciones implementadas durante la investigación (Dascalu et al., 2017)

Se concluyó finalmente que no será posible alinear perfectamente la herramienta iSTART entre ambos lenguajes, particularmente con respecto a los algoritmos de calificación. Por ello los sistemas de puntuación utilizados en la investigación deben basarse en distintos espacios de LSA que consisten en distintas fuentes de texto. Además, mejorar el pre procesamiento para el español crea diferencias sustanciales entre ambos algoritmos. (Dascalu et al., 2017).

Lexical and Syntactic Features Selection for an Adaptive Reading Recommendation System Based on Text Complexity (Kurdi, 2017)

Consulta: 15/04/2019

En esta investigación se busca construir un clasificador que pueda identificar la complejidad de una serie de textos en servicio de los alumnos del *inglés como Segundo Lenguaje* (ESL en inglés), para presentar a los alumnos los que son más adecuados para su nivel de inglés, utilizando grupo de características que pueden describir mejor la complejidad sintáctica y léxica de un texto dado. Se busca lograr esto en base a la construcción del clasificador y para su integración en un ITS enfocado a desarrollar la comprensión lectora en alumnos de ESL.

En el experimento el autor busca utilizar nuevas características como la frecuencia de palabra en el lenguaje, tiempos verbales o porcentaje de conectores de discurso que no han sido usados. Para ello se utilizaron textos educativos de páginas profesionales gratuitas para la generación del corpus.

Lo primero que se realizó fue encontrar la frecuencia de las palabras en el lenguaje, esto provee información de cuán común una palabra es, y se utilizó como un indicador para el nivel de vocabulario. Cuanto más común son las palabras en un texto, más fácil es el texto. Para este paso se removieron los stop *words* usando la lista de la herramienta NLTK, también se filtraron los nombres propios utilizando el etiquetador de NLTK, para luego realizar el proceso de *stemming* utilizando la herramienta Porter Stemmer. Cabe resaltar que esto no elimina las formar irregulares (como *make* o *made*) ya que un aprendiz del idioma debería conocer el plural simple de los sustantivos, pero no necesariamente las formas irregulares de algunos sustantivos y verbos.

Siguiente a esto se pasó a analizar la diversidad de los lemas, teniendo como hipótesis que un texto avanzado tendría una cantidad más amplia de lemas que uno básico. Realizando los procedimientos de *lemmatizing* se encontró que la correlación de la diversidad de los lemas es positiva pero significantemente menor que los criterios aplicados anteriormente (frecuencia de palabras y porcentaje de conectores de discurso).

Para obtener las características sintácticas se utilizó la herramienta Stanford Parser, para el análisis gramatical y las herramientas de NLTK como el *sentence tokenizer* y el *part-of-speech*(*POS) tagger*.

Con respecto a los tiempos verbales se implementó un módulo que usa expresiones regulares de secuencias de *POS tags* para reconocer 13 distintas formas verbales, obteniendo una precisión de 0.9 y un recall de 0.92, indicando que la mayor parte de los errores son debido a problemas con los *POS taggers*. Analizando la correlación de las distintas formas verbales se encontró que solo cinco tienen una correlación significativa. El presente simple presenta una correlación negativa ya que es el más común en textos de bajo nivel.

Explorando otras características sintácticas se menciona que estudios previos intentaron capturara distintos aspectos de la complejidad semántica a través del uso de modelos basados en n-gramas. Lo que se propone en la investigación es utilizar una cuenta simple de bi-gramas, tri-gramas, cuatro-gramas como parte de secuencias de discurso en una oración. Cuanto más diversificadas están las secuencias de etiquetas, y más complejas son las estructuras sintácticas, se puede inferir que el texto es más avanzado.

Phenomenon	Correlation
Mean number of bi-grams per sentence	-0.01
Mean number of tri-grams per sentence	0.11
Mean number of four-grams per sentence	0.20
Mean lengths of sentences	0.76

Ilustración 6: Correlación de las características de estructura sintáctica general (Kurdi, 2017)

Como se observa en la ilustración 6, solo las medias de los cuatro-gramas tienen una correlación estadística significante con la complejidad del texto, por lo que se puede decir que la longitud de una oración es un buen indicador de la complejidad de su estructura, así como de la diversidad de palabras usadas en ella.

Posteriormente, se construyeron dos modelos clasificadores con dos conjuntos distintos de características, usando cuatro algoritmos populares de ML: *Support Vector Machine* (SVM), *random forest*, *naïve Bayes* y *decision tree*, utilizando un muestreo aleatorio con tres repeticiones.

Luego de comparar los resultados se evidencia que el modelo utilizando SVM supera a los demás algoritmos para esta tarea, esto debido a su capacidad de generalizar y evitar caer en máximos/mínimos locales. Así se concluye que, si bien el rendimiento del modelo generado es prometedor, aún es posible mejorarlo. A parte de utilizar un conjunto de datos más grande, un posible camino de mejora podría ser explorar otras características sintácticas como las T-unit (unidad terminable mínima del lenguaje). (Kurdi, 2017).

Machine Comprehension with Syntax, Frames, and Semantics (Wang, Bansal, Gimpel, & McAllester, 2015)

Consulta: 16/04/2019

En esta investigación se considera el conjunto de datos de *Machine Comprehension of Text dataset* (Richardson, Burges, & Renshaw, 2013), un conjunto de historias ficticias hechas por personas con preguntas de opción múltiple.

Se explora el uso de *dependency syntax*, *frame semantics*, *word embeddings* y *coreference* para mejorar los resultados en el conjunto de datos, se menciona que las técnicas de *frame semantics y coreference* son esenciales para entender *quién hizo qué a quién*. Para esto se utilizó un clasificador de variable latente entrenado con un criterio de margen máximo. En otras palabras, teniendo una función $h: (P,q) \rightarrow A$ que, dado un pasaje de texto P y una pregunta q, retorna un elemento $a \in A$. El modelo linear para h utiliza una variable latente w para identificar la oración en el pasaje de texto en donde la respuesta puede ser encontrada. Se generan vectores de

características y vectores de peso con una entrada por cada característica para el conjunto de *f (P, w, q, a)* a fin de poder seleccionar la mejor respuesta.

Se inició con dos características de Richardson et al. (2013), la primera correspondía a medir la superposición de palabras ponderadas entre la bolsa de palabras (*bag of words, BOW*) construida de las preguntas/respuestas y la bolsa de palabras en la pregunta realizada. La segunda corresponde a la distancia entre palabras, que es la mínima distancia entre dos ocurrencias de la palabra en el pasaje que está también contenido en el par de pregunta/respuesta.

Se utilizó el analizador de marco semántico SEMAFOR (Das, Chen, Martins, Schneider, & Smith, 2014), para realizar la extracción de los predicados del marco específico en las oraciones (ver ilustración 7).

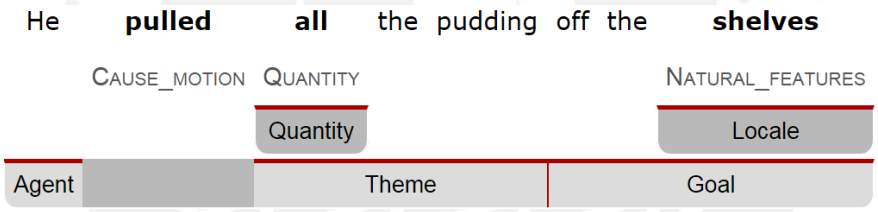


Ilustración 7: Ejemplo de resultado obtenido por SEMAFOR (Wang et al., 2015)

Como se observa en el resultado, cada estructura posee su propio conjunto de argumentos, como por ejemplo *CAUSE_MOTION* tiene la los argumentos etiquedatos *Agent*, *Theme* y *Goal*. Se menciona que las carácterísticas obtenidas de estos análisis han mostrado ser útiles para tareas de NLP.

Para comparar un par pregunta/respuesta a una oración en el pasaje, primero se usan algunas reglas para transformar la pregunta en un enunciado (ver ilustración 8) y luego insertar la respuesta candidata en la posición de rastreo.

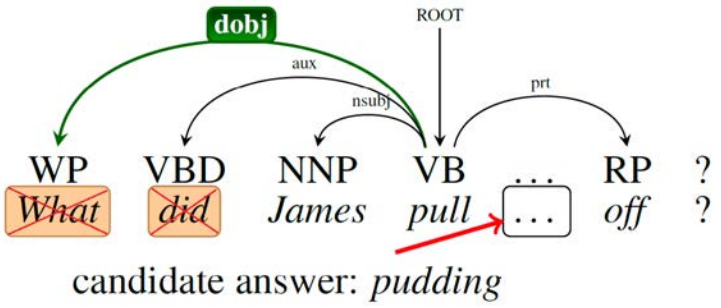


Ilustración 8: Transformación de la pregunta a un enunciado (Wang et al., 2015)

Utilizando el árbol de dependencia de Stanford y los POS tags para la pregunta se usan una serie de reglas para transformar la pregunta, por ejemplo, cambiar "why x?" a "the reason *x* is *a*". Luego de realizar dicha transformación, se mide la similitud de la oración usando simples características de similitud basadas en dependencia.

Con relación a los *word embeddings* primero se define el vector suma (f_w^+) de todas las palabras en la oración *w* y el vector multiplicación ($f_w^\times$) de los vectores dentro de la oración, luego se definen vectores para la respuesta *a* y la pregunta *q*, concatenando *q* y *a* para luego calcular los respectivos vectores suma (f_{qa}^+) y multiplicación ($f_{qa}^\times$). Para la bolsa de palabras definida, en lugar de solamente contar las ocurrencias en ambas, se utliza la igualdad de cosenos $\cos(f_{qa}^+, f_w^+)$, $\cos(f_{qa}^\times, f_w^\times)$ entre los vectores generados.

$$w = [(a, b, c), (d, e, f), \dots, (x, y, z)]$$

$$f_w^+ = [(a + d + \cdots + x), (b + e + \cdots + y) \dots (c + f + \cdots + z)]$$

$$f_w^\times = [(a \cdot d \cdot \dots \cdot x), (b \cdot e \cdot \dots \cdot y) \dots (c \cdot f \cdot \dots \cdot z)]$$

Finalmente se integra un sistema de resolución de correferencias, enfocados en identificar cadenas de menciones entre las oraciones que refieren a la misma entidad, a los métodos anteriores.

Luego de realizar los primeros experimentos se encontró que el sistema era susceptible a preguntas con negación, por lo que se desarrolló una heurística simple para solucionarlo: se

identificaban las preguntas de negación si contenían *"not"* o *"n't"* y no comenzaban con *"how"* o *"why"*. Si una pregunta era identificada como negación, entonces se negaba la puntuación final para cada pregunta candidata.

Los errores en el sistema estaban relacionados con las preguntas que requerían razonamiento inferencial, contar, establecer enumeración, oraciones múltiples, manipulación en el tiempo y comparaciones, con lo que se abre una nueva línea de investigación para mejorar este tema. Se concluye que, dado su análisis de errores encontrados en el sistema, un análisis lingüístico más profundo y la adición de razonamiento inferencial pueden generar mejoras en esta tarea. (Wang et al., 2015)

Are You Reading My Mind? Modeling Students' Reading Comprehension Skills with Natural Language Processing Techniques (Allen et al., 2015)

Consulta: 16/04/2019

En este estudio se aprovechan las herramientas de NLP para construir modelos basados en la habilidad de comprensión de los estudiantes desde las propiedades lingüísticas sobre el resultado de la aplicación de la técnica de auto-explicación. Se utilizó el sistema *Coh-Metrix* (McNamara, Graesser, McCarthy, & Cai, n.d.) para calcular las propiedades lingüísticas de las auto explicaciones, esta herramienta calcula la coherencia del texto en varias medidas distintas.

En la investigación se afirma que las interacciones con sistemas de tutoría basados en NLP conllevan a ganancias significativas de aprendizaje, comparados con tareas de aprendizaje no interactivas como leer libros o escuchar clases. Más adelante los resultados de la investigación revelan que las tareas de aprendizaje no interactivas eran equivalentes a la ausencia de todo tipo de tarea de aprendizaje.

Se utiliza el ITS iSTART como plataforma, mencionando el algoritmo de evaluación utilizado, el cuál evalúa las auto-explicaciones basadas en información de la oración objetivo del texto,

que los alumnos auto-explican, y las oraciones que rodean el pasaje. El algoritmo puntúa la calidad de la auto-explicación usando múltiples índices basados en palabras y LSA.

Para el procesamiento de los datos, se realizó una agregación de las auto-explicaciones a nivel de oraciones para cada texto que hayan leído durante su entrenamiento. Por lo tanto, cada estudiante resultó con una "auto-explicación agregada" (ver ilustración 9) para cada texto que haya leído durante su entrenamiento con iSTART. Para clarificar la idea, para un texto con *p* párrafos y *n* oraciones objetivo, la auto-explicación agregada resultante tendrá *p* párrafos y *n* auto-explicaciones correspondientes a la posición relativa de la oración objetivo.

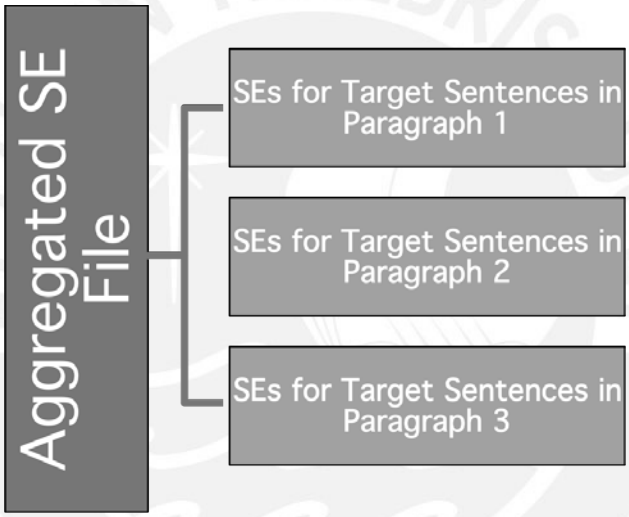


Ilustración 9: Representación visual de la auto-explicación agregada (Allen et al., 2015)

Luego, se realizó un análisis de regresión con los índices resultantes de la herramienta Coh-Metrix (McNamara et al., n.d.) como predictores sobre la puntuación de comprensión lectora de los estudiantes. El resultado del análisis indicó que los estudiantes con mayor puntaje de comprensión produjeron auto-explicaciones que contenían una mayor diversidad de palabras y oraciones más cortas. A pesar de usar una cantidad más diversa de palabras, sus auto-explicaciones contenían un mayor grado de similitud semántica que las generadas por estudiantes con mejor puntuación.

Se concluyó que los ITSs son altamente efectivos para producir un aumento en el aprendizaje entre los estudiantes, frecuentemente rindiendo los mismos resultados que tutores humanos, sin embargo, una de las mayores dificultades que se siguen enfrentando al desarrollar ITSs es la habilidad de procesar y responder a las entradas en lenguaje natural de los estudiantes. Se han empezado a utilizar técnicas de NLP con el fin de mejorar la adaptabilidad de los sistemas, si bien los algoritmos tienden a ser precisos y confiables en medir el desempeño de artículos individuales, aún se debe informar modelos más generales de las habilidades de los estudiantes. Finalmente se recalca que la diversidad lexical, cohesión semántica (LSA párrafo a párrafo) y la longitud de la oración resultaron con el mayor poder predictivo en el modelo.

3.3 Conclusiones

Dados los estudios revisados, analizando los métodos y/o algoritmos utilizados para cumplir sus objetivos, los resultados obtenidos en ellos y sus proyecciones a futuro, se puede concluir lo siguiente:

- El desarrollo de modelos de *question answering* (QA) y comprensión lectora en español aún está en una etapa primitiva, con muchos puntos por explotar y múltiples métodos que probar.
- La generación de corpus traducidos automáticamente ha presentado muchos inconvenientes para la realización de los estudios, por ello en la mayoría se optó por generarlo de manera manual o semi-automática, corrigiendo manualmente los errores. Sin embargo, mediante el análisis de calidad de las traducciones mediante las métricas a aplicar, se pretende reducir el dominio de los componentes a traducir de forma automática, para reducir el margen de error en dicho proceso.
- Se presenta la inclusión de razonamiento inferencial en los modelos para obtener un mejor resultado.

- Se presentó una mejor precisión en algoritmos de búsqueda global, como el genético o SVM, al momento de compararlos con otros algoritmos de ML.
- Finalmente, se concluye que este proyecto de fin de carrera propone una solución viable y aún no muy explorada, como es la generación automática de un corpus traducido y en base a ello implementar un modelo de comprensión lectora y question answering.

Capítulo 4. Componente de traducción de textos y programa para su evaluación y validación

4.1 Introducción

En este capítulo, se presenta el desarrollo de los resultados esperados 1, 2 y 3 del objetivo específico 1. Para este fin, primero se obtuvo el conjunto de datos inicial en inglés de la página oficial de la investigación[2]. Para el proceso de traducción automática se utilizó un servicio de *Neural Machine Translation* y posteriormente se definieron una serie de procedimientos para la evaluación de la traducción.

Para ello se realizó *backtranslation*, remoción de *stop words*, y división en *tokens*, donde un *token* representa un término o palabra parte de una oración y finalmente una comparación vectorial entre oración traducida y original por el método de Doc2Vec (Le & Mikolov, 2014).

Se aplicaron las métricas BLEU (Papineni et al., 2002) y ROUGE (Lin, 2004) para evaluar la calidad de la traducción generada.

4.2 Descripción

Se presenta un componente que posee funcionalidades para la lectura y traducción del conjunto de datos original del idioma inglés al español y posteriormente un componente de validación y pre-procesamiento.

Con respecto a la validación y evaluación se realiza el proceso de traducción inversa o *backtranslation*, haciendo uso de las métricas BLEU y ROUGE para validar la calidad de la traducción inversa, agregando luego como apoyo la similitud vectorial entre oración original y traducida haciendo uso del método Doc2Vec (Le & Mikolov, 2014). Se resalta que las métricas

[2] https://rajpurkar.github.io/SQuAD-explorer/

mencionadas anteriormente retornan un valor entre 0 y 1 el cuál indica el grado de similitud encontrado entre ambas oraciones.

4.3 Desarrollo

4.3.1 Componente de software para la lectura de datos y traducción automática de texto desde el idioma inglés al español

En la página oficial de SQuAD se pueden encontrar 2 versiones del conjunto de datos, la primera versión contando con 100,000 preguntas y respuestas en más de 500 artículos de Wikipedia y la segunda versión añadiendo más de 50,000 preguntas sin respuesta a la primera versión. Para este proyecto se utilizó la primera versión en la que todas las preguntas tienen respuesta en el texto.

El conjunto de datos viene en formato JSON, siendo estructurado de la siguiente manera:

- Artículos [lista]
    - Párrafos [lista]
        - Preguntas - Respuesta [lista]

Se desarrolló una función de lectura, la cual itera sobre cada artículo del conjunto de datos, procesando detalladamente los párrafos que lo componen y sus respectivas preguntas-respuestas. Para el caso de esta investigación, se obtuvo en total 442 artículos con un promedio de 20 párrafos y 5 preguntas por cada uno.

Una vez habiendo desarrollado la función de lectura, se implementa una función modificada para incluir la traducción automática, utilizando la API de Google Cloud Translation[3] para Python.

[3] https://cloud.google.com/translate/

Finalmente, se obtuvo un nuevo conjunto de datos estructurado de la siguiente manera:

- Artículos
    - Párrafo (original)
        - Preguntas - Respuesta (original)
    - Párrafo (traducido)
        - Preguntas - Respuesta (traducido)

De esta manera, se tomó una muestra aleatoria de 150 pares pregunta-respuesta y se los revisó manualmente para validar que no hubo inconsistencias durante el proceso de traducción automática. Luego de ello, los datos se exportaron en formato CSV para posterior procesamiento en el proyecto, así como para el entrenamiento de la línea base.

En la ilustración 10 se puede apreciar de manera gráfica el proceso de la obtención del primer resultado.

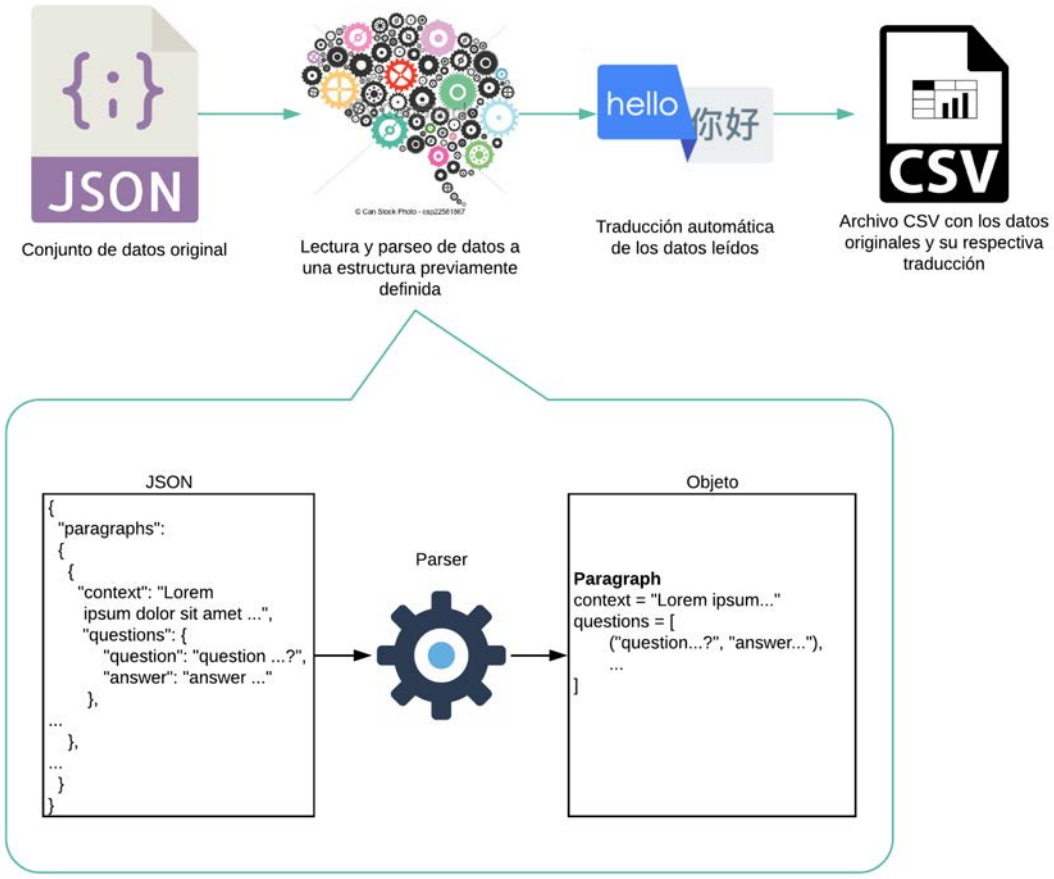


Ilustración 10: Proceso de obtención del resultado 1 [Elaboración propia]

4.3.2 Componente de pre-procesamiento de texto y cálculo de métricas para la validación automática de la traducción de acuerdo a BLEU, ROUGE y similitud vectorial semántica

La evaluación y validación se realizarán sobre el conjunto de datos traducido con el objetivo de filtrar las filas que no presenten una traducción consistente.

Esta medida se realiza mediante el método de *backtranslation*, el cual consiste en traducir de vuelta al idioma original la oración traducida, para lo cual se traducirá al inglés el párrafo en español, luego se realiza un proceso de preprocesamiento tanto al párrafo traducido de vuelta al inglés como al original.

El proceso de preprocesamiento consiste en la tokenización de ambos textos, seguido de su conversión a minúsculas. En el caso particular de esta investigación se optó por realizar la

conversión a minúsculas luego del proceso de traducción ya que hay ciertas palabras que pueden variar en la traducción debido a la presencia de mayúsculas y signos de puntuación, e.g. la palabra “Dawn” hace referencia a un nombre propio (*proper noun*) mientras que la palabra “dawn” hace referencia al sustantivo (*noun*) “amanecer”.

Para remover la puntuación se utilizó una expresión regular que realiza el filtrado de todos los tokens y seguidamente se realizó la remoción de *stop words*, utilizando el conjunto provisto por la librería NLTK.

Antes de continuar con el preprocesamiento del conjunto de datos traducido, se eliminaron las entradas en las que la respuesta a la pregunta no se encontraba textualmente en el párrafo de referencia, reduciendo el tamaño del conjunto en un 10% aproximadamente.

Respecto a la medida de similitud entre ambos textos pre procesados se utilizó la distancia entre los vectores de palabras de cada párrafo, mediante la herramienta Gensim (Řehůřek & Sojka, 2010), con los vectores de palabras entrenados en artículos de Wikipedia provistos por FastText (Bojanowski et al., 2016), dando como resultado un valor entre 0 y 1, el cual representa el grado de similitud entre los dos párrafos, donde 1 significa completamente iguales y 0 completamente distintos.

Dado que este método resultó ser muy trivial e inexacto de acuerdo a pruebas manuales realizadas sobre el modelo utilizado, se descartó el uso de los vectores de palabras. Ya que al realizar la ponderación de los vectores de la oración se pierde totalmente el contexto, se decidió entrenar un modelo de vectores de párrafos utilizando el método doc2vec (Le & Mikolov, 2014) incluido en la herramienta Gensim. Para ello se descargó el corpus oficial de los artículos de Wikipedia en inglés, con el que se entrenó un modelo de vectores de párrafo para calcular la similitud entre dos pasajes de texto o documentos calculando la distancia coseno. Cabe resaltar que, a diferencia del método descrito anteriormente, toma en cuenta la frecuencia de

las palabras en cada documento y el contexto de la palabra para calcular la distancia entre ambos vectores. El resultado es un valor entre 0 y 1 indicando el grado de similitud entre ambos documentos, donde 0 es completamente distintos y 1 completamente iguales.

Para el entrenamiento de los vectores de párrafo se aplica el método *Distributed Bag of Words*, el cuál considera la concatenación del vector de párrafo (D en la ilustración 11) con los vectores de palabra calculados de los tokens de la oración/párrafo para predecir la siguiente palabra en una ventana de texto y no tiene en cuenta el orden de las palabras en su entrenamiento (Le & Mikolov, 2014).

En la ilustración 11 se detalla el modelo utilizado para el entrenamiento de los vectores de párrafo sobre artículos de Wikipedia.

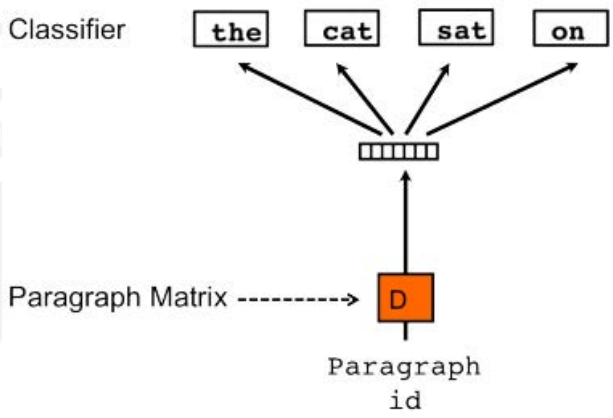
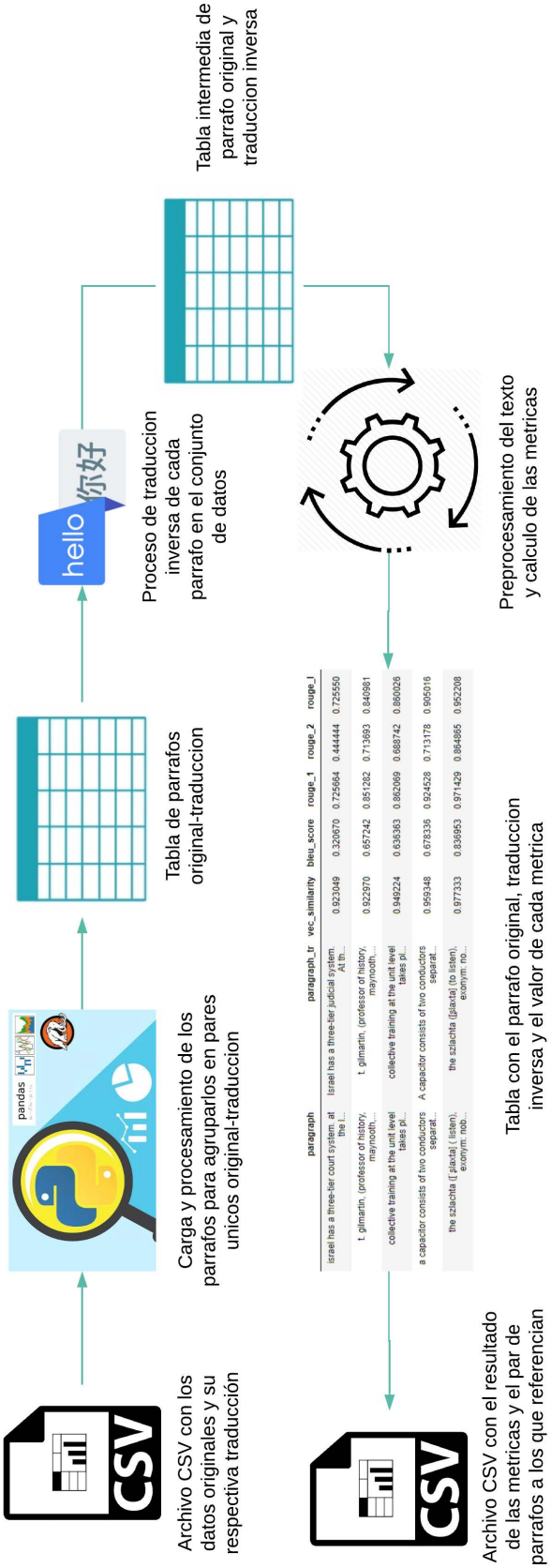


Ilustración 11: Modelo utilizado para el entrenamiento de los vectores de párrafos (Le & Mikolov, 2014)

Se utiliza también la métrica BLEU (Papineni et al., 2002) para medir la similitud entre la oración original y la oración resultante de la traducción inversa. No se utilizó para medir la precisión del modelo de traducción, que es para lo que normalmente se aplica esta métrica, sino se empleó su forma de evaluar oraciones de acuerdo a la ocurrencia de términos y n-gramas de la oración a analizar contra la oración de referencia. A la par se implementó la métrica ROUGE (Lin, 2004), la cual funciona de manera similar que BLEU (Papineni et al., 2002), enfocándose

en la ocurrencia de unigramas y bigramas de la oración de referencia respecto a la oración a analizar. En la ilustración 12 se muestra el proceso relacionado la obtención de este resultado.

Ilustración 12: Proceso de obtención del resultado 2 [Elaboración propia]



4.3.3 Experimentación numérica de las métricas calculadas sobre un proceso de *backtranslation* para el procesamiento del conjunto de datos

Se realiza un análisis de las distribuciones de cada métrica con respecto a los valores agrupados en rangos de 0.1 y la distribución de los valores de manera continua.

Se realiza el análisis de las métricas BLEU (Papineni et al., 2002) y ROUGE (Lin, 2004) al inicio ya que, al ser métricas diseñadas para medir la ocurrencia de n-gramas entre dos textos utilizando una variación de la precisión y *recall* respectivamente, son las que tienen una mayor influencia en el criterio de filtrado y se adaptan mejor al tipo de validación que se quiere realizar. Teóricamente, el texto original y el texto producto de *backtranslation* debe ser el mismo, pero en la práctica no es así, por lo que se requiere validar que tan cercano es el contenido de ambos textos. Se presentan como métricas complementarias.

La ilustración 13 muestra un diagrama de barras de frecuencias agrupadas por rangos, el cual muestra la cantidad de pares de textos que obtuvieron un valor de la métrica BLEU en los intervalos indicados de 0.1, mientras que la ilustración 14 muestra la distribución de los valores de la métrica en forma de histograma.

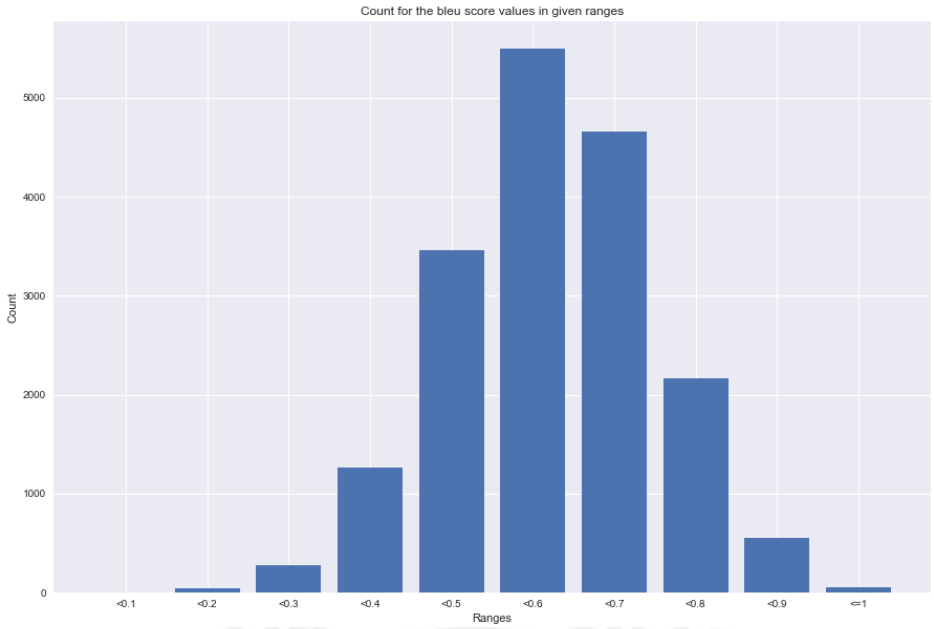


Ilustración 13: Gráfico de barras de la métrica BLEU en intervalos de 0.1

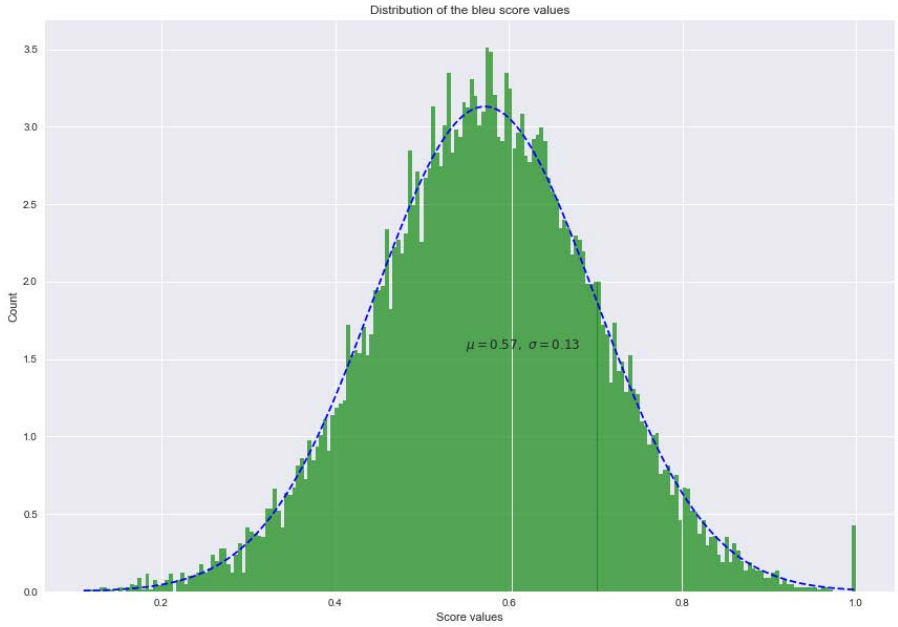


Ilustración 14: Histograma de la distribución de valores de la métrica BLEU

En el histograma se puede observar que los valores poseen un amplio rango, acumulándose la mayor cantidad cerca a la media, lo que puede ser confirmado por la desviación estándar.

Las gráficas confirman que la métrica BLEU tiene una alta importancia en la validación de la traducción inversa y es sensible a las variaciones entre los textos.

Con respecto a la métrica ROUGE, se aplicaron dos versiones de la métrica: Rouge-1 y Rouge-2, siendo ambas basadas en unigramas y bigramas, respectivamente.

En la ilustración 15 se muestra un gráfico de barras que muestra la cantidad de pares de textos que obtuvieron un valor de la métrica ROUGE-1 en los intervalos indicados de 0.1.

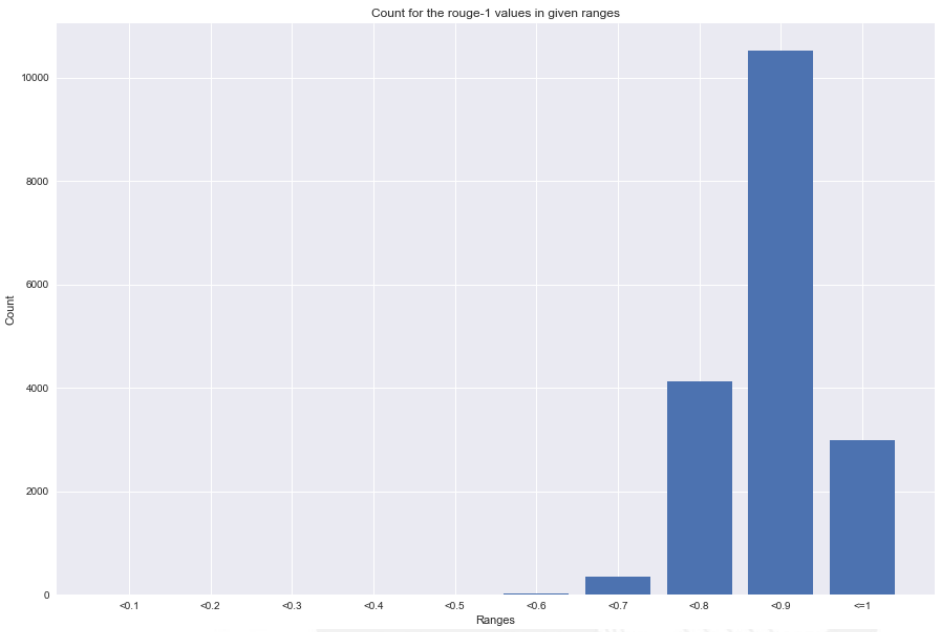


Ilustración 15: Grafico de barras de la métrica ROUGE-1 en intervalos de 0.1

La ilustración 16 muestra la distribución de los valores de la métrica ROUGE-1 en forma de histograma, mostrando una distribución menos dispersa que BLEU y con una media mucho más alta. Lo cual se puede comprobar observando la desviación estándar.

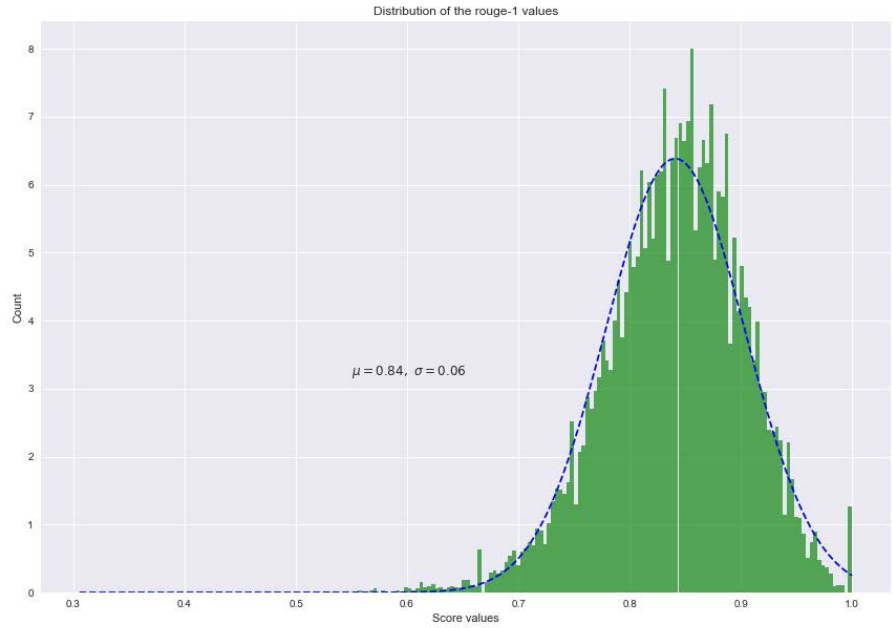


Ilustración 16: Histograma de la distribución de valores de la métrica ROUGE-1

En la ilustración 17 se muestra un gráfico de barras que muestra la cantidad de pares de textos que obtuvieron un valor de la métrica ROUGE-2 en los intervalos indicados de 0.1.

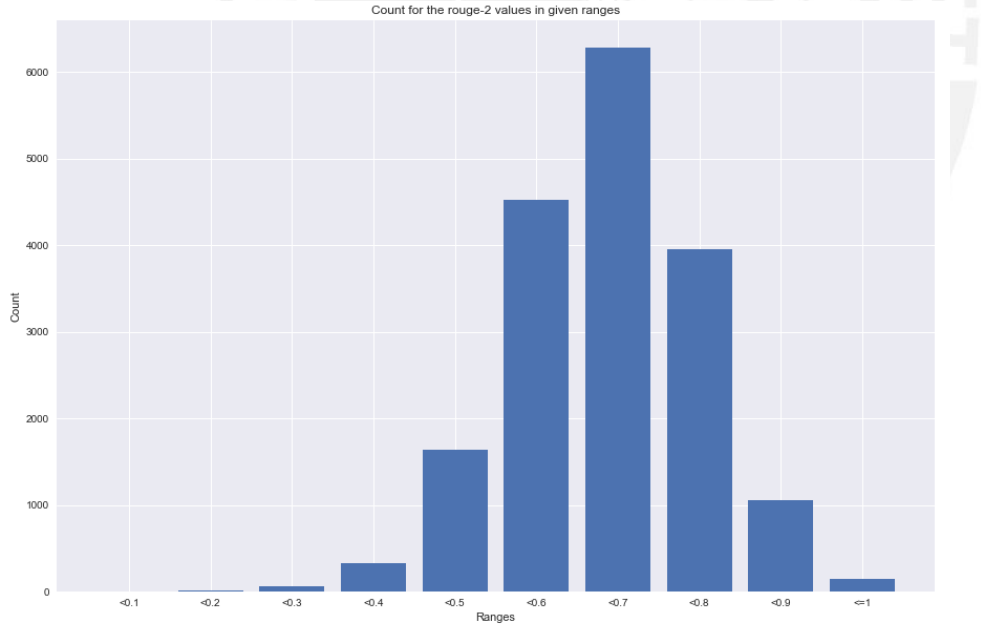


Ilustración 17: Grafico de barras de la métrica ROUGE-2 en intervalos de 0.1

La ilustración 18 muestra la distribución de los valores de la métrica ROUGE-2 en forma de histograma, mostrando una distribución con valores más dispersos y con un mayor rango de

valores que la variación anterior, esto se debe a que ahora se mide el *recall* de los bigramas en el texto de referencia respecto al texto de la traducción inversa.

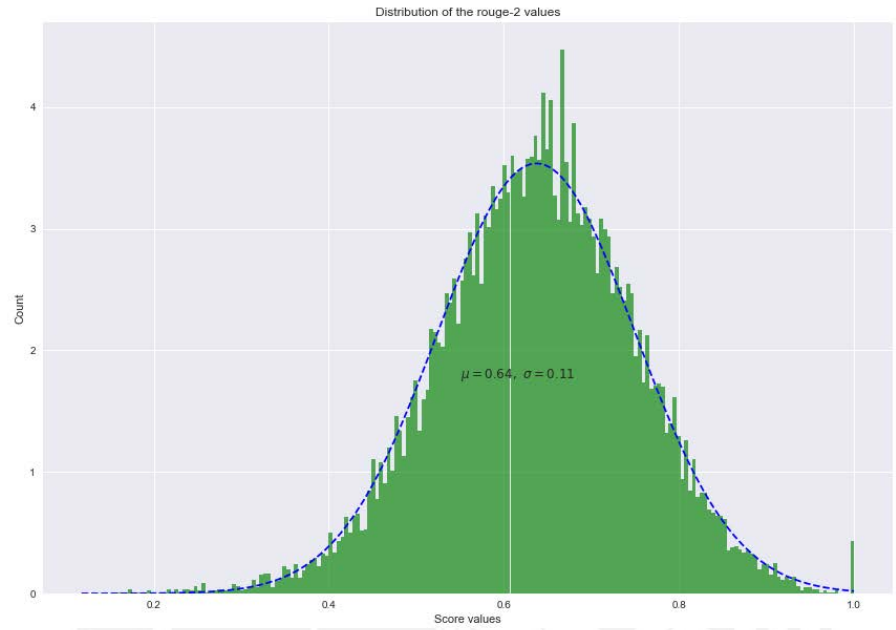


Ilustración 18: Histograma de la distribución de valores de la métrica ROUGE-2

Por otra parte, en la ilustración 19 se muestra un gráfico de barras que muestra la cantidad de pares de textos que obtuvieron un valor de distancia coseno respecto a sus vectores de documento en los intervalos indicados de 0.1.

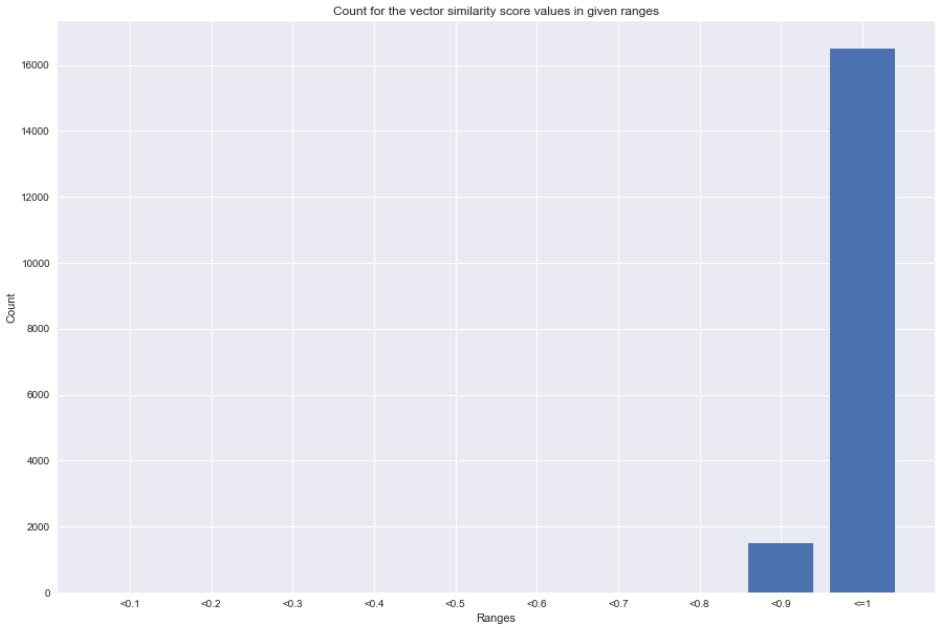


Ilustración 19: Grafico de barras de la distancia coseno entre vectores de documento en intervalos de 0.1

La ilustración 20 muestra la distribución de los valores de la distancia coseno respecto a sus vectores de documento en forma de histograma, el cual muestra una distribución muy poco dispersa y con una media muy alta.

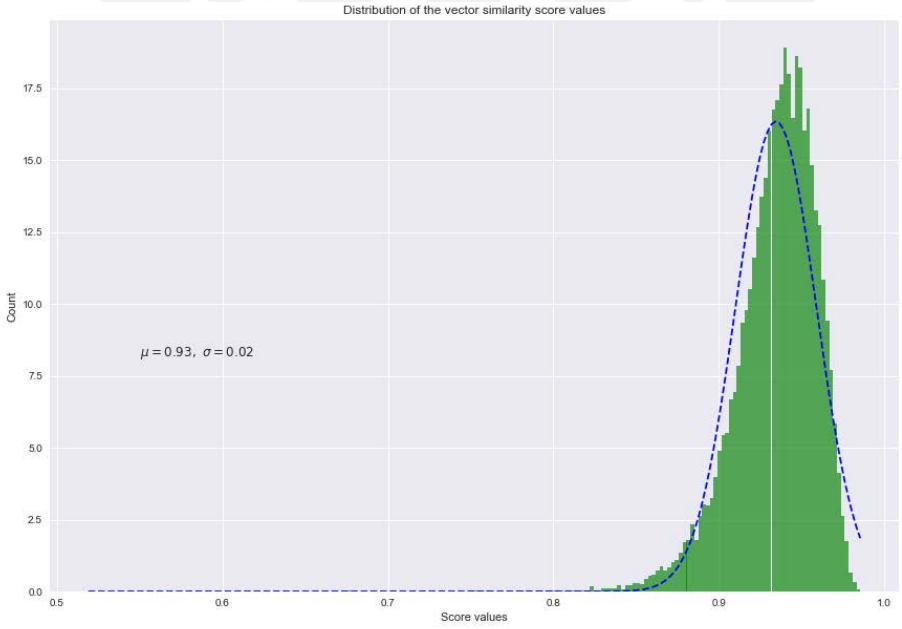


Ilustración 20:Histograma de la distribución de los valores de la distancia coseno entre vectores de documento

Analizando los gráficos se puede corroborar que las métricas BLEU y ROUGE son las que aportan mayor información para un proceso de validación y filtrado, en caso el valor de estas dos métricas no sea el óptimo, se utilizaría la distancia coseno entre los vectores de documento de los textos, los cuales tienen en cuenta el contexto a comparación de las dos primeras métricas que se basan en la ocurrencia de n-gramas.

Para comprobar que las métricas BLEU y ROUGE son complementarias, se realizó un análisis sobre una ponderación de ambas:

$$\alpha \times BLEU + (1 - \alpha) \times ROUGE, \qquad \forall \alpha \in [0,1]$$

Los valores de α serán incrementados en 0.1 desde 0 hasta 1, con lo cual se obtuvieron los siguientes resultados:

Tabla 5: Tabla de resultados de la ponderación de BLEU y ROUGE [Elaboración propia]

$\boldsymbol{\alpha}$	**Función**	**Media (μ)**	**Desviación Estándar (σ)**
0.1	$0.1 \times BLEU + (0.9) \times ROUGE$	0.63	0.11
0.2	$0.2 \times BLEU + (0.8) \times ROUGE$	0.62	0.12
0.3	$0.3 \times BLEU + (0.7) \times ROUGE$	0.62	0.12
0.4	$0.4 \times BLEU + (0.6) \times ROUGE$	0.61	0.12
0.5	$0.5 \times BLEU + (0.5) \times ROUGE$	0.6	0.12
0.6	$0.6 \times BLEU + (0.4) \times ROUGE$	0.6	0.12
0.7	$0.7 \times BLEU + (0.3) \times ROUGE$	0.59	0.12
0.8	$0.8 \times BLEU + (0.2) \times ROUGE$	0.59	0.12

0.9	$0.9 \times BLEU + (0.1) \times ROUGE$	0.58	0.13

En la ilustración 21 se muestra el F-Score promedio para cada valor de α, obteniendo una gráfica bastante simétrica con su punto más alto cuando el valor de α fue 0.5, con esto se puede comprobar que tanto BLEU como ROUGE tienen la misma importancia al momento de validar el par de textos.

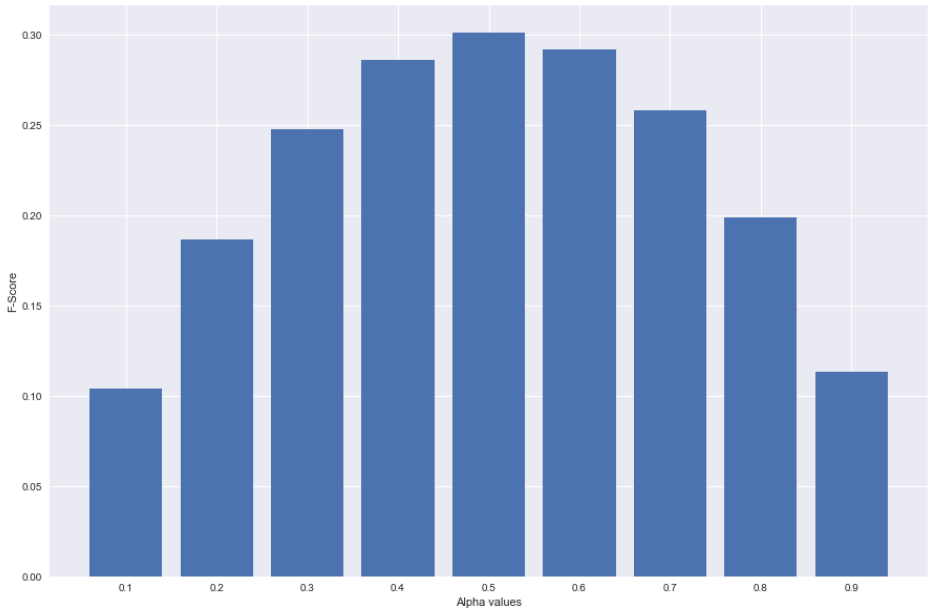


Ilustración 21: F-Score promedio por cada valor de α

Finalmente, se realiza el filtrado del conjunto de datos priorizando BLEU y ROUGE, y teniendo la similitud vectorial como una métrica de ayuda. Se definió como punto de corte una desviación estándar antes de la media para todas las métricas.

En la ilustración 22 se muestra el criterio utilizado en el proceso de filtrado, teniendo en cuenta las jerarquías entre las métricas.

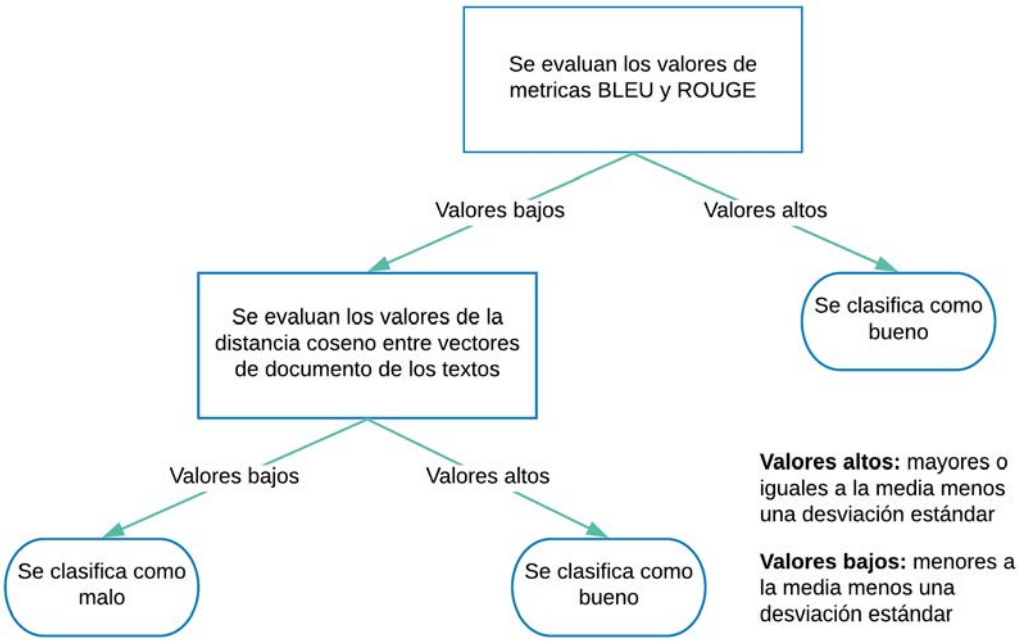


Ilustración 22: Proceso de filtrado del conjunto de datos traducido [Elaboración propia]

Capítulo 5. Implementación de un modelo algorítmico de comprensión lectora en español

5.1 Introducción

En este capítulo se presentan los resultados esperados 4 y 5, relacionados al objetivo específico 2. Para el desarrollo de este objetivo se utilizó la estructura del modelo algorítmico de redes neuronales propuesto en la investigación *Reading Wikipedia to Answer Open-Domain Questions* (Chen et al., 2017) adaptado a los conjunto de datos obtenidos en el objetivo anterior.

5.2 Descripción

Se presentan dos modelos entrenados sobre dos conjuntos de datos traducidos, uno sin procesar y otro procesado por los métodos descritos en el capítulo anterior.

El modelo algorítmico implementado hace uso de redes neuronales recurrentes multicapa entrenado para detectar respuestas en los párrafos/contextos del conjunto de datos. El modelo algorítmico realiza un pre-procesamiento sobre el conjunto de datos para transformarlo en un formato que pueda ser procesado correctamente. En esta parte se transforman las preguntas y contextos en una lista de *tokens*, a la par también se utilizan vectores de palabras para construir un vocabulario y realizar un emparejamiento entre las palabras del conjunto de datos y el vocabulatio que se tiene, de esta manera formando una matriz *word_to_id* donde cada palabra es asignada un identificador (id) que representa su posición en el vocabulario. Finalmente se utilizan estos métodos para definir la capa de *embeddings* de la red neuronal, la cuál utilizará los vectores de palabras para transformar los *inputs* en su forma vectorial.

5.3 Desarrollo

5.3.1 Modelo de línea base entrenado sobre el conjunto de datos inicial traducido, sin procesamiento

Se entrena el modelo algorítmico descrito con el conjunto de datos traducido, sin filtrado ni procesado por las métricas calculadas para el resultado 2.

Se toma un porcentaje del conjunto de entrenamiento y el conjunto de validación para formar un conjunto de prueba, el cual será utilizado para evaluar todos los modelos algorítmicos que se entrenen. Se busca que el conjunto de prueba tenga un tamaño lo más cercano posible al conjunto de validación (siendo el caso de esta investigación es un aproximado de 7,000 ejemplos), el cual representa ~10% del conjunto de entrenamiento original. Obteniendo finalmente tres conjuntos nuevos: uno de entrenamiento, uno de validación y otro de prueba (que se utilizará como conjunto de prueba general).

Una vez definido el conjunto de prueba se realiza el pre-procesamiento de los nuevos conjuntos de entrenamiento y validación para adaptarlos al formato de entrada de la red neuronal recurrente. Este proceso consiste en generar el vocabulario a partir de los vectores de palabras utilizados. En el caso de esta investigación se utilizaron vectores GloVe (Pennington, Socher, & Manning, 2014) entrenados en el idioma español, para posteriormente generar una matriz de *word_to_id* en donde cada palabra del vocabulario contenida en los conjuntos de entrenamiento/validación es asignada a un índice que corresponde a su posición en el vocabulario generado de los vectores.

Se guarda el índice, contexto, pregunta, respuesta, posición de inicio y fin de la respuesta en 6-tuplas, en el caso del conjunto de entrenamiento. Para el conjunto de validación se guarda el índice, contexto, pregunta y respuesta en 4-tuplas.

Toda esta información procesada es separada y guardada en dos archivos utilizando el formato *MessagePack* por motivos de eficiencia. El primer archivo conteniendo los datos del conjunto de entrenamiento y validación, llámese el archivo "*data.msgpack"* y el segundo archivo que contiene los vectores y las matrices de *word_to_id*, llámese el archivo *"meta.msgpack"*.

Estos dos archivos son leídos por el método de entrenamiento del modelo algorítmico, para el cuál se definieron los siguientes parámetros basándose en la investigación citada al inicio de este capítulo:

- Número de épocas: 40
- *Batch size*: 32
- Dimensión de los embeddings: 300
- Optimización: SGD (Stochastic Gradient Descent)
- Learning rate: 0.1

Respecto a la evaluación de las predicciones, se utilizaron las dos métricas recomendadas para evaluar sistemas de comprensión lectora, las cuales ignoran puntuación y artículos (Jurafsky & Martin, 2018):

- *Exact Match (EM) score*: El porcentaje de predicciones que son exactamente iguales a las respuestas esperadas.
- *F1 score*: La superposición media entre las respuestas predichas y las respuestas esperadas. Se trata la predicción y el esperado como una bolsa de *tokens*, y se calcula el F1, promediando el F1 sobre todas las preguntas.

Ambas métricas retornan un valor entre 0 y 1, siendo 0 que no se predijo ninguna respuesta correctamente, según la métrica, y 1 que se predijo la totalidad del conjunto correctamente.

Luego de realizar el entrenamiento del modelo algorítmico de comprensión lectora con un conjunto de entrenamiento de ~57,000 ejemplos y un conjunto de validación de ~7,000

ejemplos, se obtuvo un EM score de 49.18% y un F1 score de 61.97% sobre el conjunto de validación.

Posteriormente se realizó el cálculo de las métricas sobre el conjunto de prueba con 6363 ejemplos, obteniendo un EM score de 47.28% y un F1 score de 61.88%. De esta manera, se comprueba que el modelo algorítmico obtenido no se sobre entrenó y brinda predicciones coherentes en un rango esperado.

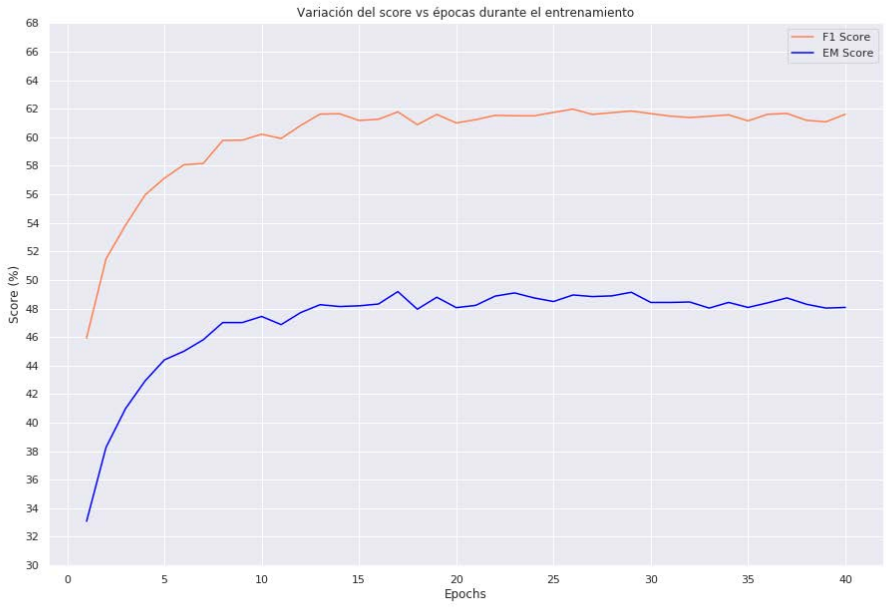


Ilustración 23: Variación de F1 y EM scores por épocas durante el entrenamiento del modelo con datos traducidos sin procesar

5.3.2 Modelo algorítmico final entrenado sobre el conjunto de datos traducido y procesado

Se entrena el modelo algorítmico descrito con el conjunto de datos traducido y filtrado de acuerdo a las métricas y criterio definido en el resultado 3 teniendo el mismo conjunto de prueba que el resultado anterior.

El conjunto de datos de entrenamiento será de un tamaño menor al utilizado en el resultado anterior ya que se aplicó un punto de corte de acuerdo a los valores obtenidos de las métricas

de validación de calidad de la traducción. En el caso del presente proyecto el conjunto de datos filtrado contiene ~33,000 ejemplos, un 42% más pequeño.

Los métodos de pre-procesamiento son los mismos que se utilizaron en el resultado anterior, al igual que los parámetros definidos para el entrenamiento del modelo algorítmico y las métricas para la validación de las predicciones (EM y F1).

Luego de realizar el entrenamiento del modelo algorítmico de comprensión lectora con un conjunto de entrenamiento de ~33,000 ejemplos y un conjunto de validación de ~5,000 ejemplos, se obtuvo un EM score de 45.53% y un F1 score de 58.81% sobre el conjunto de validación.

Posteriormente, al igual que el modelo algorítmico del resultado anterior, se calcularon las métricas sobre el conjunto de prueba con 6363 ejemplos, obteniendo un EM score de 40.95% y un F1 score de 55.61%. Obteniendo buenos resultados, pero no tan precisos como el modelo algorítmico del resultado anterior (línea base).

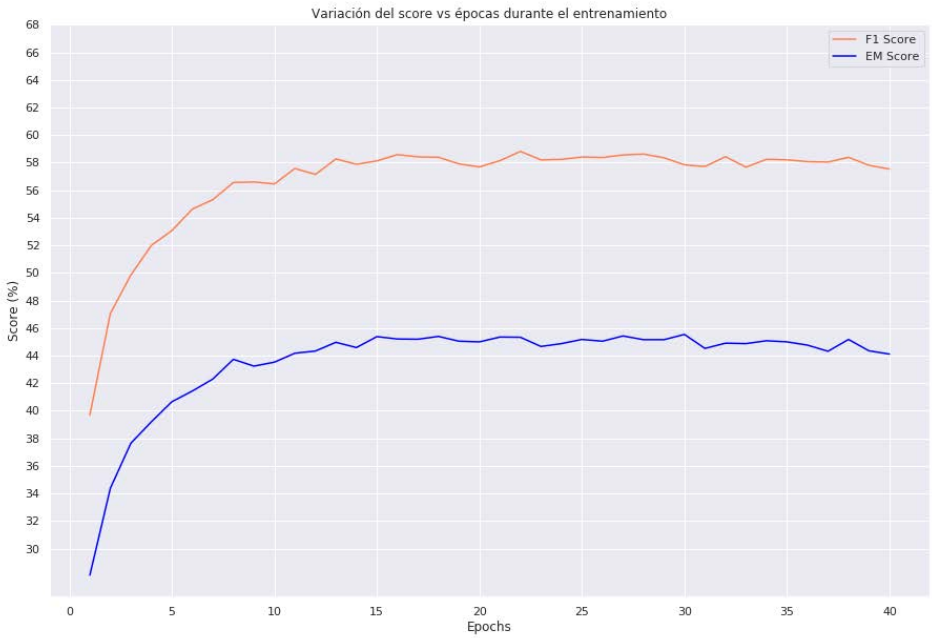


Ilustración 24: Variación de F1 y EM scores por épocas durante el entrenamiento del modelo con datos traducidos procesados

5.4 Discusión de resultados

Comparando los resultados obtenidos por ambos modelos algorítmicos, se observa que el modelo entrenado con el conjunto de datos sin procesar obtuvo una mayor puntuación que el entrenado con el conjunto de datos procesado, lo cual contradice a lo planteado en la hipótesis inicial, la cual establece que un conjunto de datos procesado y filtrado utilizando métricas automáticas de calidad de traducción es esperado a aumentar el rendimiento de un modelo algorítmico de comprensión lectora con respecto a uno entrenado con un conjunto de datos traducido sin procesar.

Un factor que se considera determinante en el resultado obtenido es la antigüedad de las métricas utilizadas, las cuales se utilizaron en su versión clásica. La posible baja precisión de las métricas al procesar los datos lleva a obtener un conjunto de datos final reducido en un 41%, lo cual impacta en la calidad del modelo algorítmico obtenido ya que los modelos de redes neuronales obtienen mejores resultados con mayor cantidad de datos.

Una medida propuesta para mejorar este resultado es utilizar los estudios realizados en estos dos últimos años sobre la mejora de estas métricas de evaluación de traducción automática, como Meteor++(Guo, Ruan, & Hu, 2018) o (Fomicheva, Uria Bel, Specia, & Malinovskiy, 2016), las cuales hacen referencia al problema encontrado en el desarrollo del objetivo y exploran alternativas plausibles para mejorar la validación de la traducción automática.

En la Tabla 6, se pueden observar los resultados entre los dos modelos entrenados y la comparación entre ellos, además de la comparación con el modelo original en idioma nativo.

Tabla 6: Tabla de resultados obtenidos por los modelos algorítmicos entrenados [Elaboración propia]

Resultado		***F1 Score* (%)**	***EM Score* (%)**

	Ejemplos de entrenamiento	*Validation*	*Test*	*Validation*	*Test*
1	57,232	61.97	61.88	49.18	47.28
2	33,537	58.81	55.61	45.53	40.95
Modelo original (Chen et al., 2017)	88,013	78.9		69.4	

Capítulo 6. Componente de software para encapsular y presentar las funcionalidades del modelo algorítmico implementado

6.1 Introducción

En este capítulo se presenta el resultado esperado 6, relacionado al objetivo específico 3. Para el desarrollo de este objetivo se utilizó el modelo algorítmico que obtuvo el mejor F1 score, para posteriormente definir un servicio REST utilizando la librería Flask (Ronacher, 2010) para Python, definiendo finalmente una interfaz web simple para facilitar la interacción con el resultado del proyecto.

6.2 Descripción

Se presenta una Interfaz de Programación de Aplicaciones (API) la cuál recibe un contexto y una pregunta y utiliza el modelo algorítmico de comprensión lectora desarrollado para predecir la respuesta a la pregunta planteada sobre el contexto y posteriormente la retorna en formato JSON. Esta API está alojada en un servidor con los recursos necesarios para ejecutar el modelo algorítmico de comprensión lectora, ejecutándose como un servicio REST en una dirección pública, la cual puede ser consultada desde cualquier dispositivo con acceso a internet.

Para facilitar su uso y probar el funcionamiento, se decidió realizar una interfaz web simple, la cual se conecta internamente con el servicio y permite la interacción con el modelo desde cualquier dispositivo con acceso a internet.

6.3 Desarrollo

6.3.1 Interfaz de Programación de Aplicaciones (API)

Se crea un nuevo script de Python importando la librería Flask (Ronacher, 2010) la cual permitirá crear un servicio REST que escuche a nuevas peticiones de predicciones sobre un contexto y pregunta dadas.

En la estructura de la API a implementar se define un método que será el encargado de cargar en memoria el modelo generado en el objetivo anterior, con los métodos provistos por la librería PyTorch (Paszke et al., 2017), al momento de iniciar el servicio.

La estructura del servicio consiste en un solo método POST el cuál recibe como parámetros el contexto y la pregunta realizada, retornando una respuesta en formato JSON incluyendo la predicción y un *flag* que indica si la petición se realizó con éxito.

Este servicio es ejecutado en un puerto específico, al cuál se puede acceder mediante la dirección IP pública del servidor que aloja dicho servicio.

La estructura de consulta al servicio es la siguiente:

http://<IP_SERVER>**:**<PORT>**/predict?evidence=**<CTX>**&question=**<QS>

Siendo

- **IP_SERVER**, la dirección IP pública del servidor que aloja el servicio
- **PORT**, el puerto en el que se esta ejecutando el servicio
- **CTX**, el pasaje de texto que se desea analizar
- **QS**, la pregunta sobre el texto provisto

La consulta debe ser realizada por el método POST del protocolo HTTP y retorna una respuesta en formato JSON de la siguiente manera dependiendo de si se tuvo éxito procesando la consulta o no:

```
{
   “evidence”: <CTX>,
   “question”: <QS>,
   “answer”: <ANS>,
   “success”: true
}
```

```
{
        “success”: false
}
```

Donde **CTX** y **QS** representan el pasaje de texto y la pregunta, respectivamente; **ANS** representa la predicción obtenida por el modelo algorítmico y ***“success”*** hace referencia a un *flag* que indica el éxito de la ejecución de la consulta. En caso que la consulta no sea exitosa se retornará solo el *flag* en false.

Finalmente, en el caso de esta investigación, para probar el correcto funcionamiento de la API implementada, se realizó una simple interfaz web la cual se conecta internamente con el servicio, siguiendo el formato definido líneas arriba, la cual puede ser consultada en la dirección http://spanishqa.ml

Ilustración 25: Interfaz de usuario para el ingreso de datos

Ilustración 26: Interfaz de usuario para la muestra de resultados

Capítulo 7. Conclusiones y trabajos futuros

En esta sección se presentan las conclusiones a las que se llegó y los trabajos futuros planteados luego del desarrollo del presente proyecto.

7.1 Conclusiones

- Es posible generar un conjunto de datos automáticamente traducido del idioma inglés al español y entrenar un modelo algorítmico de comprensión lectora con un desempeño aceptable, basándose en los resultados obtenidos en los capítulos 4 y 5, midiéndolos contra la línea base en el idioma nativo: *Reading Wikipedia to Answer Open-Domain Questions* (Chen et al., 2017).
- Las métricas propuestas para la validación de la traducción y el consecuente conjunto de datos filtrado afectaron el desempeño del modelo de comprensión lectora, esto genera varias posibilidades de análisis y en este caso se enfocará en dos:
    - El criterio definido para el filtrado de las traducciones fue en base a métricas clásicas que fueron propuestas varios años atrás siendo, de esta manera, no muy exactas para los procedimientos que se realizan en la actualidad.
    - Debido a que las métricas generaron la reducción del conjunto de datos en aproximadamente un 40%, reduciendo considerablemente la cantidad de ejemplos a usar en el entrenamiento del modelo de comprensión lectora lo cual afectó su desempeño final ya que este tipo de modelos requieren grandes cantidades de datos.

  El primer filtro realizado luego de la traducción, que consiste en eliminar las preguntas en las cuales su respuesta no se encontraba textualmente en el párrafo de referencia (4.3.2), también afectó la precisión del modelo ya que se eliminaron varios ejemplos en este proceso. El impacto de este filtro está relacionado directamente la efectividad de

la traducción automática y puede ser reducido utilizando un método más complejo de traducción automática o curando manualmente los ejemplos que presenten esta peculiaridad (respuestas no contenidas en el párrafo de referencia).

- El modelo algorítmico de comprensión lectora utiliza satisfactoriamente los ejemplos vistos en el entrenamiento para responder de manera correcta a un porcentaje significativo de las preguntas que se le realiza sobre un pasaje de texto dado, basándose en los *scores* obtenidos por el modelo base en idioma nativo (Tabla 6).

7.2 Trabajos futuros

- Mejorar el proceso de validación de la traducción automática y *backtranslation* utilizando nuevas métricas que proponen un desempeño superior a las métricas clásicas de calidad como BLEU y ROUGE. Estos son los casos de CobaltF (Fomicheva et al., 2016), Meteor++ (Guo et al., 2018), RUSE (Shimanaka, Kajiwara, & Komachi, 2018) e ITER (Panja & Naskar, 2018), con las cuales se espera obtener una validación más robusta y precisa con el objetivo de mejorar el desempeño del modelo de comprensión de lectura. Cabe resaltar que en la conferencia de *machine translation* 2018 (WMT18) se presentó una estudio que mejoró el valor de la métrica BLEU obtenida en un sistema de traducción automática al filtrar su conjunto de datos (Bei et al., 2018). Lo que refuerza la hipótesis que realizar el correcto filtrado del conjunto de datos traducido mejora los resultados obtenidos.
- Analizar la alternativa sobre la implementación un mejor modelo de traducción automática al utilizado y su relación con la posible mejora del modelo algorítmico de comprensión lectora.

Referencias

Allen, L. K., Snow, E. L., & McNamara, D. S. (2015). Are you reading my mind? In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge - LAK '15* (pp. 246–254). New York, New York, USA: ACM Press. https://doi.org/10.1145/2723576.2723617

Bar-Hillel, Y. (1960). A Demonstration of the Nonfeasibility of Fully Automatic High Quality Translation, *1*. Retrieved from http://www.mt-archive.info/Bar-Hillel-1960-App3.pdf

Bei, C., Zong, H., Wang, Y., Fan, B., Li, S., & Yuan, C. (2018). An Empirical Study of Machine Translation for the Shared Task of WMT18, *2*, 344–348. https://doi.org/10.18653/v1/W18-64031

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python*. O'Reilly.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching Word Vectors with Subword Information. Retrieved from http://arxiv.org/abs/1607.04606

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J. D., … Roossin, P. S. (n.d.). A STATISTICAL APPROACH TO MACHINE TRANSLATION. Retrieved from http://www.aclweb.org/anthology/J90-2002

Chen, D., Bolton, J., & Manning, C. D. (2016). A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. Retrieved from http://arxiv.org/abs/1606.02858

Chen, D., Fisch, A., Weston, J., & Bordes, A. (n.d.). *Reading Wikipedia to Answer Open-Domain Questions*. Retrieved from http://trec.nist.gov/data/qamain.html

Chen, D., Manning, C., Bolton, J., Fisch, A., Bordes, A., & Weston, J. (2017). Towards the Machine Comprehension of Text. Retrieved from

https://cs.stanford.edu/people/danqi/talks/berkeley_20170410.pdf

Chowdhury, G. G. (2005). Natural language processing. *Annual Review of Information Science and Technology*, *37*(1), 51–89. https://doi.org/10.1002/aris.1440370103

Das, D., Chen, D., Martins, A. F. T., Schneider, N., & Smith, N. A. (2014). Frame-Semantic Parsing. *Computational Linguistics*, *40*(1), 9–56. https://doi.org/10.1162/COLI_a_00163

Dascalu, M., Jacovina, M. E., Soto, C. M., Allen, L. K., Dai, J., Guerrero, T. A., & McNamara, D. S. (2017). Teaching iSTART to Understand Spanish (Vol. 10331 LNAI, pp. 485–489). Springer Verlag. https://doi.org/10.1007/978-3-319-61425-0_46

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From Data Mining to Knowledge Discovery in Databases. Retrieved from https://www.aaai.org/ojs/index.php/aimagazine/article/viewFile/1230/1131

Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., … Welty, C. (2010). Building Watson: An Overview of the DeepQA Project. *AI Magazine*, *31*(3), 59. https://doi.org/10.1609/aimag.v31i3.2303

Fomicheva, M., Uria Bel, N. ´, Specia, L., & Malinovskiy, A. (2016). *CobaltF: A Fluent Metric for MT Evaluation* (Vol. 2). Retrieved from https://github.com/amalinovskiy/

Gaspers, J., Karanasou, P., & Chatterjee, R. (2018). Selecting Machine-Translated Data for Quick Bootstrapping of a Natural Language Understanding System. Retrieved from https://arxiv.org/pdf/1805.09119.pdf

Google LLC. (n.d.). Google Cloud Translation API. Retrieved June 8, 2018, from https://cloud.google.com/translate/

Guo, Y., Ruan, C., & Hu, J. (2018). Meteor++: Incorporating Copy Knowledge into Machine Translation Evaluation, *2*, 753–758. https://doi.org/10.18653/v1/W18-64082

Hewlett, D., Lacoste, A., Jones, L., Polosukhin, I., Fandrianto, A., Han, J., … Berthelot, D. (2016). WikiReading: A Novel Large-scale Language Understanding Task over Wikipedia. Retrieved from http://arxiv.org/abs/1608.03542

Hill, F., Bordes, A., Chopra, S., & Weston, J. (2015). The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations. Retrieved from http://arxiv.org/abs/1511.02301

Honnibal, M. (2015). Introducing spaCy. Retrieved June 1, 2018, from https://explosion.ai/blog/introducing-spacy

Jurafsky, D., & Martin, J. H. (2017). Question Answering. In *Speech and Language Processing (3rd ed. draft)* (3rd ed.). Retrieved from https://web.stanford.edu/~jurafsky/slp3/28.pdf

Jurafsky, D., & Martin, J. H. (2018). *Speech and Language Processing Question Answering*. Retrieved from https://web.stanford.edu/~jurafsky/slp3/23.pdf

Kurdi, M. Z. (2017). Lexical and syntactic features selection for an adaptive reading recommendation system based on text complexity. In *ACM International Conference Proceeding Series* (Vol. Part F1282, pp. 66–69). https://doi.org/10.1145/3077584.3077595

Kwok, C., Etzioni, O., & Weld, D. S. (2001a). Scaling question answering to the web. *ACM Transactions on Information Systems*, *19*(3), 242–262. https://doi.org/10.1145/502115.502117

Kwok, C., Etzioni, O., & Weld, D. S. (2001b). Scaling question answering to the web. *ACM Transactions on Information Systems*, *19*(3), 242–262. https://doi.org/10.1145/502115.502117

Le, Q., & Mikolov, T. (2014). *Distributed Representations of Sentences and Documents*. Retrieved from https://arxiv.org/pdf/1405.4053.pdf

Liddy, E. D. (2001). Natural Language Processing. In *Encyclopedia of Library and Information Science* (2nd ed.). NY: Marcel Decker, Inc. Retrieved from https://surface.syr.edu/cgi/viewcontent.cgi?referer=https://scholar.google.com.pe/&httpsredir=1&article=1019&context=cnlp

Lin, C.-Y. (2004). *ROUGE: A Package for Automatic Evaluation of Summaries*. Retrieved from http://www.aclweb.org/anthology/W/W04/W04-1013.pdf

Mckinney, W. (2011). pandas: a Foundational Python Library for Data Analysis and Statistics. Retrieved from http://www.dlr.de/sc/Portaldata/15/Resources/dokumente/pyhpc2011/submissions/pyhpc2011_submission_9.pdf

McNamara, D. S., Graesser, A. C., McCarthy, P. M., & Cai, Z. (n.d.). *Automated evaluation of text and discourse with Coh-Metrix*. Retrieved from http://www.cambridge.org/zw/academic/subjects/psychology/educational-psychology/automated-evaluation-text-and-discourse-coh-metrix#y7wdgX0eulhz5HKk.97

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012a). *Foundations of Machine Learning*. MIT Press. Retrieved from https://books.google.com.pe/books?id=-ijiAgAAQBAJ&dq=what+is+machine+learning&lr=&source=gbs_navlinks_s

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012b). *Foundations of Machine Learning*. MIT Press.

Monsalve, F., Rivas-Rojas, K., Antonio, M., Cabezudo, S., & Oncevay, A. (2019). *Assessing Back-Translation as a Corpus Generation Strategy for non-English Tasks : A Study in Reading Comprehension and Word Sense Disambiguation*. *The 13th Linguistic Annotation Workshop*.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., & Deng, L. (2016). MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. Retrieved from http://arxiv.org/abs/1611.09268

Onishi, T., Wang, H., Bansal, M., Gimpel, K., & McAllester, D. (2016). Who did What: A Large-Scale Person-Centered Cloze Dataset. Retrieved from https://arxiv.org/abs/1608.05457

Panja, J., & Naskar, S. K. (2018). ITER: Improving Translation Edit Rate through Optimizable Edit Costs, *2*, 759–763. https://doi.org/10.18653/v1/W18-64083

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, Q. N., Bernardi, R., Pezzelle, S., … Fernández, R. (2016). The LAMBADA dataset: Word prediction requiring a broad discourse context. Retrieved from http://arxiv.org/abs/1606.06031

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). *BLEU: a Method for Automatic Evaluation of Machine Translation*. Retrieved from http://aclweb.org/anthology/P/P02/P02-1040.pdf

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., … Lerer, A. (2017). Automatic differentiation in PyTorch. In *NIPS 2017 Workshop Autodiff Program Chairs*. Retrieved from https://openreview.net/pdf?id=BJJsrmfCZ

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, É. (2012). Scikit-learn: Machine Learning in Python. Retrieved from https://arxiv.org/abs/1201.0490

Pennington, J., Socher, R., & Manning, C. D. (2014). *GloVe: Global Vectors for Word Representation*. Retrieved from http://nlp.

Pérez Zorilla, J. (2005). EVALUACIÓN DE LA COMPRENSIÓN LECTORA: DIFICULTADES Y LIMITACIONES. *Revista de Educación*, 121–138. Retrieved from

http://114.red-88-12-10.staticip.rima-tde.net/mochila/sec/monograficos_sec/ccbb_ceppriego/lengua/aspgenerales/M Jesus Perez.pdf

Project Jupyter. (2015). Retrieved May 31, 2018, from http://jupyter.org/about

Psotka, J., Massey, L. D. (Leonard D., & Mutter, S. A. (1988). *Intelligent tutoring systems : lessons learned*. L. Erlbaum Associates.

Python Software Foundation. (2001). Retrieved May 31, 2018, from https://www.python.org/psf-landing/

Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). SQuAD: 100,000+ Questions for Machine Comprehension of Text. Retrieved from http://arxiv.org/abs/1606.05250

Řehůřek, R., & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (pp. 45–50). Valletta, Malta: ELRA.

Richardson, M., Burges, C. J. C., & Renshaw, E. (2013). MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text, 193–203. Retrieved from https://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/MCTest_EMNLP2013.pdf

Ronacher, A. (2010). Flask (A Python Microframework). Retrieved June 8, 2018, from http://flask.pocoo.org/

Schraagen, J. M., Chipman, S. F., & Shalin, V. L. (2000). *Cognitive Task Analysis*. Taylor & Francis. Retrieved from https://books.google.com.pe/books?id=gyt5AgAAQBAJ

Searle, J. (2011). John Searle: Watson Doesn't Know It Won on Jeopardy! Retrieved April 25, 2018, from

https://www.wsj.com/articles/SB10001424052748703407304576154313126987674

Shimanaka, H., Kajiwara, T., & Komachi, M. (2018). RUSE: Regressor Using Sentence Embeddings for Automatic Machine Translation Evaluation, *2*, 764–771. https://doi.org/10.18653/v1/W18-64084

Somers, H. (2005). Round-Trip Translation: What Is It Good For?, 127–133. Retrieved from http://www.mt-archive.info/ALTW-2005-Somers.pdf

Trischler, A., Wang, T., Yuan, X., Harris, J., Sordoni, A., Bachman, P., & Suleman, K. (2016). NewsQA: A Machine Comprehension Dataset. Retrieved from http://arxiv.org/abs/1611.09830

Vanlehn, K. (2011). The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. *Educational Psychologist*, *46*(4), 197–221. https://doi.org/10.1080/00461520.2011.611369

Wang, H., Bansal, M., Gimpel, K., & McAllester, D. (2015). Machine comprehension with syntax, frames, and semantics. In *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference* (Vol. 2, pp. 700–706).

Waskom, M. (2012). Seaborn: statistical data visualization. Retrieved May 31, 2018, from https://seaborn.pydata.org/

Yampolskiy, R. V. (n.d.). Turing Test as a Defining Feature of AI-Completeness. *Artif. Intell. Evol. Comput. and Metaheuristics SCI*, *427*, 3–17. Retrieved from http://cecs.louisville.edu/ry/TuringTestasaDefiningFeature04270003.pdf

ET LUX IN TENEBRIS LUCET
MCMXVII

Anexo 1

	Actividad	Duracion planificada (dias)	Inicio	Fin	Dependencia
1	Traduccion del conjunto de datos original al español	7	15-08-2018	21-08-2018	-
2	Revision manual a una muestra aleatoria del conjunto de datos traducido	3	22-08-2018	24-08-2018	1
3	Aplicacion de metricas de calidad de la traduccion como BLEU	5	25-08-2018	29-08-2018	1
4	Implementacion de los metodos de validacion de la traduccion (backtranslation) y preprocesamiento del texto	7	30-08-2018	05-09-2018	2
5	Entrenamiento de la linea base con el conjunto de datos traducido sin validar	7	06-09-2018	12-09-2018	2
6	Entrenamiento de modelos con el conjunto de datos final	15	13-09-2018	27-09-2018	4
7	Medicion y prueba de los resultados obtenidos luego del entrenamiento de los modelos	5	28-09-2018	02-10-2018	6
8	Modificacion o mejora del modelo seleccionado	7	03-10-2018	09-10-2018	7
9	Pruebas al modelo en situaciones reales	7	10-10-2018	16-10-2018	8
10	Desarrollo de una API que encapsule las funciones del modelo generado	12	17-10-2018	28-10-2018	9
11	Desarrollo de una interfaz de usuario web que haga uso de la API generada	10	29-10-2018	08-11-2018	10