UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

Facultad de Ingeniería

Generación de preguntas y respuestas para comprensión lectora en inglés utilizando modelos neuronales

---

Autores:

Tatiana Rischewski
Gonzalo Berger

Supervisores:

Aiala Rosá
Luis Chiruzzo

3 de mayo de 2022

Resumen

Dentro del marco del proyecto “Inglés sin Límites” del Programa de Políticas Lingüístcas de ANEP, se propone el desarrollo de una herramienta orientada al apoyo de la enseñanza de inglés en distintos centros estudiantiles del país. Esta herramienta debe ser capaz de realizar la generación de preguntas y respuestas automática a partir de un texto, con el objetivo de la confección de evaluaciones. Para esto, se propone un enfoque neuronal, en el que se entrenan modelos de generación de preguntas que, a partir de respuestas previamente seleccionadas de un texto, generen una serie de preguntas de un nivel de inglés A1/A2.

Lo que proponemos en este trabajo es la separación de la tarea de generación de preguntas y respuestas en dos etapas independientes, una de selección de respuestas a partir de un texto, y otra de generación de preguntas a partir de un conjunto de respuestas. En la etapa de selección de respuestas se determina un conjunto de etiquetas de rol semántico de PropBank y luego se utiliza semantic role labeling para identificar palabras en el texto que coincidan con alguna de las etiquetas del conjunto y así poder extraerlas como respuestas candidatas. En la etapa de generación de preguntas se tienen modelos neuronales basados en T5 y GPT2, preentrenados sobre SQuAD y NewsQA, que producen una colección de preguntas en base a los candidatos a respuestas obtenidos de la etapa anterior.

Se propone también la adición de dos etapas adicionales, además de las dos originales: una de pre y otra de postprocesamiento. En la etapa de preprocesamiento se aplica coreference resolution sobre el texto de entrada, y en la de postprocesamiento se resuelven algunas malformaciones que pueden presentar las preguntas generadas. Estas etapas fueron agregadas al observar las salidas obtenidas antes de implementarlas, y podemos concluir que son importantes para obtener mejores resultados y un mejor desempeño en general de la herramienta final.

En cuanto a los resultados obtenidos, se obtuvo que los modelos entrenados basados en T5 superan ampliamente a aquellos entrenados sobre GPT-2. La herramienta final, utilizando uno de los modelos entrenados sobre T5, presenta buenos resultados sobre el corpus específico para evaluación, que contiene textos de un nivel de inglés A1/A2. Por otro lado, en comparación con otros trabajos previos de generación de preguntas utilizando redes neuronales, obtenemos resultados competitivos en un corpus de uso general como es SQuAD. Debido a esto podemos concluir que la herramienta desarrollada presenta un buen desempeño en la tarea de generación de preguntas y además es capaz de generar preguntas del nivel de inglés buscado.

Índice

1. Introducción

La enseñanza de inglés es un foco fundamental en la educación actual, tanto en el país como en el mundo, al punto que muchas personas comienzan con este aprendizaje en una etapa muy temprana de su vida: la escuela. En una gran parte de las escuelas del país la enseñanza de inglés ya está consolidada, es algo que ya es parte íntegra de los proyectos de enseñanza.

Sin embargo, no siempre se dispone de los recursos o capacidades para proveer de una buena enseñanza de inglés, como es principalmente el caso de las escuelas rurales. Estas escuelas muchas veces carecen de personal capacitado para la enseñanza de inglés, dado que estos centros estudiantiles presentan una falta de accesibilidad que complica la posibilidad de tener un profesor presencial, y una falta de conectividad que impide la enseñanza por videoconferencia. Para atacar este problema es que surge el proyecto "Inglés sin Límites" del Programa de Políticas Lingüístcas de ANEP[3], que busca que la enseñanza de inglés no dependa de la presencia de un docente ni de la conexión a internet.

Con el fin de que este programa prospere es que es necesario el desarrollo de herramientas que permitan automatizar ciertas etapas del proceso de enseñanza del idioma. Una posible ayuda que podría facilitar las condiciones de enseñanza sería el desarrollo de una herramienta que facilite a los maestros la tarea de la enseñanza de inglés, por ejemplo, que automatice el proceso de generación de evaluaciones, con tal de que los maestros solamente deban contrastar las respuestas de los estudiantes con las generadas con la herramienta. De esta manera se puede conseguir que los maestros puedan realizar evaluaciones aún sin un amplio conocimiento del idioma.

Lo que se propone en este trabajo es el desarrollo de una herramienta que permita generar de forma automática preguntas y respuestas a partir de un texto dado. Este proyecto sigue fuertemente la línea de trabajo del proyecto de grado realizado por Joaquín Scocozza Díaz y Martín Jaime Morón en 2019 [34], donde también se desarrolló una herramienta de la misma índole. La principal diferencia entre estos dos trabajos es que en ese caso se utilizó un enfoque basado en reglas (las preguntas-respuestas son generadas a partir de una serie de reglas específicas) mientras que en el presente trabajo se utiliza un enfoque basado en redes neuronales (se entrenan modelos para que luego generen las preguntas-respuestas automáticamente).

La motivación principal del grupo fue trabajar sobre un proyecto directamente vinculado con la enseñanza. La escasez de herramientas y la pobre infraestructura de enseñanza de inglés que hay en muchos centros de educación implica una limitación en la educación de muchos niños y jóvenes, y la posibilidad de desarrollar una herramienta que ayude a mitigar esta limitación fue algo fundamental para la elección del proyecto.

Además, el Procesamiento de Lenguaje Natural es un área de interés en el grupo, y la posibilidad de profundizar sobre técnicas y conceptos para el desarrollo de una herramienta, además de llevar los conceptos a un caso de uso práctico en la vida real, fueron también factores influyentes en la motivación del grupo.

El objetivo general del trabajo es el desarrollo de la herramienta de generación automática de preguntas, con el fin de facilitar la tarea de evaluación de inglés en contextos donde la infraestructura para su enseñanza no sea la adecuada.

En cuanto a los objetivos específicos del trabajo, se identificaron los siguientes:

- Continuar con la línea de trabajo del proyecto de grado de 2019, desarrollando una herramienta de la misma índole pero ofreciendo un enfoque basado en redes neuronales.
- Realizar un estudio de trabajos relacionados de generación de preguntas utilizando también un enfoque neuronal.
- Entrenar distintos modelos para la generación de preguntas.
- Desarrollar una herramienta para extraer respuestas a partir de un texto.
- Desarrollar un corpus de preguntas y respuestas del nivel específico de inglés sobre el que se trabaja (niveles A1 y A2).
- Evaluar distintas implementaciones de la herramienta, por ejemplo utilizando distintos modelos de lenguaje para la generación de preguntas.
- Profundizar en el área del Procesamiento de Lenguaje Natural mediante la exploración de distintas herramientas, estudiando trabajos relacionados y adquiriendo conocimiento sobre conceptos nuevos y otros ya conocidos.

El documento se estructura de la siguiente manera:

En el capítulo 2 se presentan los conceptos relevantes y las técnicas y herramientas del Procesamiento de Lenguaje Natural utilizados a lo largo del proyecto. También se describen distintos trabajos relacionados, entre los que se incluye el proyecto de 2019, que ayudan a mostrar el contexto y dar una base para el trabajo realizado.

En el capítulo 3 se describen con detalle los distintos corpus utilizados para el desarrollo de la herramienta. Primero se detallan corpus existentes y comúnmente utilizados en trabajos similares, como SQuAD y NewsQA, y luego se describe el corpus específico generado para este trabajo, que consiste de la adaptación del corpus construido por los autores del proyecto de 2019, junto con otros textos y sus correspondientes pares de preguntas y respuestas, anotados y generados dentro del marco de este proyecto.

En el capítulo 4 se presentan los distintos aspectos de la herramienta desarrollada. Se muestra el pipeline de la generación de preguntas a partir de un texto de entrada y se explica detalladamente cada una de sus etapas, así como las decisiones que llevaron a incluir cada una de ellas.

En el capítulo 5 se muestran los resultados obtenidos por cada uno de los modelos entrenados, así como una comparación entre algunos de estos modelos con las herramientas desarrolladas en otros trabajos relacionados.

En el capítulo 6 se listan las conclusiones obtenidas, se evalúa el cumplimiento de los objetivos planteados, y se presentan distintas líneas de trabajo futuro que se pueden seguir para lograr una mejoría de la herramienta.

2. Revisión de antecedentes

En esta sección se presentan distintos conceptos necesarios para la comprensión de este documento, así como algunos recursos utilizados en la realización del mismo.

También se presenta un resumen de trabajos previos relacionados y del estado del arte en el marco de generación de preguntas y respuestas.

2.1. Marco teórico

2.1.1. PLN

El Procesamiento del Lenguaje Natural (PLN) es una subdisciplina de la Inteligencia Artificial que intenta resolver mediante computadoras tareas vinculadas al lenguaje humano, permitiendo la comunicación entre el humano y la computadora a través del lenguaje natural o resolviendo tareas que implican algún tipo de procesamiento de texto o habla.[23]

Durante mucho tiempo los sistemas de procesamiento de lenguaje natural se basaron en conjuntos de reglas escritas a mano, hasta que se introdujeron algoritmos de aprendizaje automático. En este documento nos centramos en este último enfoque, construyendo sobre el proyecto de 2019[34] que se desarrolló utilizando la primera alternativa.

Algunas de las tareas principales de PLN son la traducción automática, la extracción y recuperación de información, el resumen automático de textos, el reconocimiento de voz, la generación de textos y la respuesta de preguntas[12]. Para este trabajo lo que se desea es generar preguntas con sus respectivas respuestas a partir de un texto de entrada. Para ello se utilizan de manera conjunta la extracción de información (para obtener respuestas candidatas a partir de un texto) y la generación de texto (para generar una pregunta a partir de una respuesta candidata y su contexto). Para la primera tarea se hace uso del proceso de etiquetado de roles semánticos (semantic role labeling), que se introducirá más adelante, mientras que para la segunda se utilizan modelos de lenguaje entrenados sobre conjuntos de datos pertinentes.

2.1.2. Semantic role labeling

El etiquetado de roles semánticos o semantic role labeling es una tarea de PLN que consiste en asignar etiquetas a palabras o frases dentro de oraciones que indican qué rol representan dentro de la misma. Esencialmente se busca encontrar qué elementos de la oración responden a preguntas como “quién”, “qué”, “dónde” o “cuándo”. [24]

El etiquetado de roles semánticos puede ser útil en cualquier aplicación de procesamiento de lenguaje natural que requiera comprensión semántica, como puede ser traducción, extracción de información, resúmenes de textos y más. En este trabajo, como veremos en la sección 4.2, se utiliza para extraer candidatos a respuestas de distintos textos de entrada.

Algunos de los roles son:

Agente El causante voluntario de un evento.

$$[\textbf{El mozo}]_{AGENTE} \text{ derramó la sopa.}$$

Experimentante El testigo o experimentador de un evento.

$$[\textbf{Juan}]_{EXPERIMENTANTE} \text{ tiene dolor de cabeza.}$$

Fuerza El causante no voluntario de un evento.

$$[\textbf{El viento}]_{FUERZA} \text{ abrió la ventana.}$$

Tema El mayor afectado por el evento.

$$\text{María rompió } [\textbf{el hielo}]_{TEMA}.$$

Resultado El resultado de un evento.

$$\text{El obrero construyó } [\textbf{un muro}]_{RESULTADO}.$$

Contenido La proposición o contenido de un evento.

$$\text{Carolina dijo } [\textbf{“¿Conocés a Juan?”}]_{CONTENIDO}.$$

Instrumento Un instrumento utilizado en un evento.

$$\text{Juan abrió la puerta } [\textbf{con la llave}]_{INSTRUMENTO}.$$

Beneficiario El beneficiario de un evento.

$$\text{Pedro compró el boleto } [\textbf{para su jefe}]_{BENEFICIARIO}.$$

Origen El origen de un evento.

$$\text{Volé } [\textbf{desde Chile}]_{ORIGEN}.$$

Destino El destino de un evento.

$$\text{Manejé } [\textbf{hasta Las Piedras}]_{DESTINO}.$$

En este trabajo también utilizamos la notación utilizada por PropBank, un corpus de textos anotados con información sobre proposiciones semánticas básicas. Este corpus es de uso común como información de entrenamiento en el desarrollo de sistemas de etiquetado de roles semánticos basados en *machine learning*.[26][35][19]

PropBank denota a los argumentos de un verbo como argumentos enumerados (ARG0, ARG1, ARG2, etc.), y además, PropBank agrega etiquetas particulares para algunos modificadores del verbo. Estos modificadores pueden ser de modo, de lugar o de tiempo. La notación planteada por PropBank y los roles representados por cada etiqueta se pueden ver en la Tabla 1.[9]

ARG0	Agente	ARG3	Origen, Beneficiario, Atributo
ARG1	Experimentante	ARG4	Destino
ARG2	Instrumento, Beneficiario, Atributo	ARGM	Modificador

Tabla 1: Notación de Propbank

ARGM se extiende como:

- ARGM-MNR: Modificador de modo.

  Juan habló con Maria **a solas**$_{ARGM-MNR}$, en su casa, el Jueves.

- ARGM-LOC: Modificador de lugar.

  Juan habló con Maria a solas, **en su casa**$_{ARGM-LOC}$, el Jueves.

- ARGM-TMP: Modificador de tiempo.

  Juan habló con Maria a solas, en su casa, **el Jueves**$_{ARGM-TMP}$.

Cabe destacar que ARG0 y ARG1 siempre representan esos roles, mientras que los ARG del 2 al 4 varían su rol dependiendo del predicado.

2.1.3. Coreference resolution

La resolución de correferencias (o *coreference resolution* en inglés) es la tarea de PLN que trata de encontrar todas las expresiones que refieren a la misma entidad dentro de un texto. A este conjunto de expresiones se le llama cadena o *cluster* de correferencias. Es una tarea importante para otros procesos de procesamiento de lenguaje natural como el resumen de textos, *question answering*, o extracción de información.[24]

Los sistemas modernos de resolución de correferencias están basados en aprendizaje automático neuronal, entrenados sobre datasets etiquetados manualmente. Uno de estos modelos es el que implementa AllenNLP[28], siendo el que utilizamos en este trabajo.

Utilizamos AllenNLP para realizar un preprocesamiento del texto de entrada, para que no incluya pronombres u otras palabras que refieran a nombres propios, de manera que en los pasos siguientes se obtengan mejores resultados. Para ello, se sustituyen las palabras pertenecientes al mismo *cluster*, por la primera palabra del *cluster* (primera palabra del *cluster* que se encuentra en el texto). Por ejemplo, la siguiente oración:

Adriana is tall. She has brown hair. Mathew is eleven. He likes basketball.

Tiene como clusters: *[Adriana, She], [Mathew, He]*. Sustituyendo se transforma en la siguiente oración:

Adriana is tall. Adriana has brown hair. Mathew is eleven. Mathew likes basketball.

2.1.4. Named entity recognition

El reconocimiento de entidades nombradas (named entity recognition o NER en Inglés), es una tarea de PLN que busca encontrar *spans* de texto que representen nombres propios, y luego etiquetarlos con el tipo de entidad correspondiente. Los cuatro tipos de entidad más comunes son PER (persona), LOC (lugar), ORG (organización) y GPE (entidad geopolítica). Además, las entidades nombradas pueden también incluir fechas, horas e incluso expresiones numéricas como precios. [24] El siguiente ejemplo muestra algunas de estas etiquetas:

Pedro$_{PER}$ viajó a **Canadá**$_{LOC}$.

En el proyecto de 2019[34] esta fue una de las tareas utilizadas para el procesamiento de texto de entrada, junto a las mencionadas *coreference resolution* y *semantic role labeling*.

2.1.5. Redes Neuronales

Las redes neuronales artificiales son sistemas computacionales, en particular son un subconjunto del aprendizaje automático (machine learning en inglés), algoritmos que pueden mejorar su desempeño sobre una tarea automáticamente mediante el uso de datos. [32]

Las redes neuronales están compuestas por un conjunto de unidades o nodos interconectados, comúnmente llamados neuronas, que reciben una entrada y generan una salida que se envía a otras neuronas. Cada neurona realiza una suma ponderada de todas sus entradas, ponderando de acuerdo a pesos que se le asignan a las conexiones, que luego pasa a una función de activación para obtener la salida.

Por lo general se pueden organizar en capas donde cada neurona recibe información de la capa anterior. La capa que recibe información del exterior se llama capa de entrada, la capa que produce el resultado final es la capa de salida, y las capas intermedias se llaman capas ocultas. Deep learning se refiere a aprendizaje automático con redes neuronales profundas, donde más de 3 capas se puede considerar profundo. [5]

Existen diferentes tipos de redes neuronales, dependiendo de cómo se estructuren las capas, junto con sus entradas y salidas. El más sencillo son las redes feed-forward, donde cada nodo de una capa considera como parte de la entrada a las salidas de la capa anterior. La información fluye desde la entrada, atravesando capa a capa, hasta la salida. No hay retroalimentación entre las capas.

En contraposición a estas están las redes neuronales recurrentes (RNN), en donde la salida de un nodo puede afectar a una capa anterior. La salida en un cierto tiempo puede actuar como parte de la entrada en un tiempo posterior. Esto hace que pueda guardar un estado interno como "memoria", lo que permite procesar datos secuenciales. [5]

Para el entrenamiento se pueden utilizar distintos esquemas de aprendizaje, dependiendo de cómo sea la información que se le provee al modelo. El más común y en el que nos centramos para el entrenamiento en este trabajo es el aprendizaje (o entrenamiento) supervisado, que es cuando se provee de un conjunto de datos de ejemplo etiquetados, a partir de estos el modelo aprende los pesos que mejoran los resultados para este conjunto y luego el modelo se encarga de hacer predicciones para entradas no etiquetadas.[33]

En este aprendizaje, lo que se busca es optimizar la función de pérdida (*loss function* o *cost function*) del modelo, y para ello, se busca realizar una optimización paso a paso. En cada paso de aprendizaje (o *step*) se actualizan los pesos según un tamaño de paso que depende de la tasa de aprendizaje (*learning rate*). Una forma de minimizar (y la que usamos para los modelos) es el denominado método de descenso por gradiente, que consiste en hacer la actualización en el sentido opuesto al gradiente de la función de pérdida con respecto a los pesos.

Existen dos variantes para la actualización de los pesos dentro del método del gradiente, el descenso por batch (*batch descent*) y el descenso estocástico (*stochastic descent*). En el enfoque por batch, se considera todo el dataset en la función de pérdida, por lo que es una variante costosa computacionalmente dado que requiere evaluaciones sobre todo el dataset. El segundo enfoque, el descenso estocástico, utiliza subconjuntos del dataset para cada paso. El tamaño de este subconjunto se conoce como *batch size*. Este enfoque es también llamado *stochastic single descent* cuando el batch size es 1 y *stochastic mini-batch descent*, cuando el batch size es mayor a 1.[25]

Estas iteraciones pueden recorrer el dataset una determinada cantidad de veces. Esto depende de la cantidad de *epochs* especificada. Un *epoch* es una recorrida sobre todo el dataset. Entonces la cantidad de epochs determina la cantidad de recorridas que se hacen sobre el dataset dentro del entrenamiento.[6]

Al entrenar un modelo, es necesario indicar los valores de distintos hiperparámetros, que afectan en gran medida al aprendizaje del propio modelo. Los hiperparámetros son aquellos parámetros que en lugar de ser determinados por el modelo a medida que va aprendiendo, son especificados como argumentos en la entrada[33]. Algunos ejemplos de hiperparámetros ya fueron mencionados en esta sección (batch size, learning rate y cantidad de epochs).

2.1.6. Transfer learning

Transfer learning es un método de aprendizaje automático donde el aprendizaje desarrollado para una tarea de aprendizaje se reutiliza como punto de partida en otra tarea de aprendizaje. [10]

Como el procesamiento del lenguaje natural es un área muy diversa, por lo general los *datasets* para cada tarea específica no son tan grandes, de solo miles o cientos de miles de ejemplos. Los modelos actuales de aprendizaje profundo se benefician de tener muchos datos para entrenar (millones o billones de ejemplos de entrenamiento). Por esto se generaron técnicas que permiten entrenar un modelo de lenguaje general con muchos datos sin anotar (*pre-training*), y luego entrenar sobre una tarea específica con un corpus más pequeño anotado (*fine-tuning*). Algunos ejemplos de este tipo de modelos pre-entrenados son BERT, GPT, ELMo, T5. [14] [41] [40] [42]

El concepto de preentrenamiento (*pretraining*) es el proceso por el cual un modelo aprende una representación del significado de palabras u oraciones al procesar grandes cantidades de texto. Llamamos *fine-tuning* al proceso de tomar estas representaciones y seguir entrenando el modelo, por lo general con un conjunto de textos más pequeño que además suele estar anotado.[24]

A los modelos que mencionamos se les suele llamar modelos de lenguaje, y se encargan de asignar una probabilidad a una tira de palabras de entrada de manera de estimar qué tan factible es que la secuencia sea un texto apropiado en un determinado idioma.

Esta probabilidad se asigna mediante una distribución de probabilidad dada y observando los elementos anteriores dentro de la misma secuencia. Un modelo n-grama es un modelo

que observa en las anteriores n palabras de la que se está procesando actualmente, así, la probabilidad de una palabra solo depende de las n anteriores. Desde un punto de vista neuronal, el equivalente a los modelos de n-grama serían los modelos unidireccionales, es decir, que solo observan los elementos previos de la secuencia. Una arquitectura que trabaja de esta forma es GPT.[41]

También existen modelos bidireccionales, que no solo observan en los elementos previos de la secuencia, sino también aquellos que vienen después. Estos modelos son comúnmente utilizados en aplicaciones de generación de discurso y de machine learning. Un ejemplo de esta arquitectura es BERT.[14]

2.1.7. Transformers

Transformers es un modelo de *deep learning* fuertemente basado en mecanismos de atención [7] presentado por Vaswani et al. [49]. Los mecanismos de atención son mecanismos en los que el modelo presta mayor atención a distintos elementos a la hora de procesar la información. Transformers utiliza la variante denominada como *self-attention*, en la que se determina la interdependencia entre elementos de la entrada. Para ello, se calcula una serie de pesos para cada elemento de la entrada que indica su importancia con respecto a los demás elementos.

El modelo Transformer sigue un esquema tradicional de *encoder-decoder*, donde tanto el *encoder* como el *decoder* consisten en un stack de seis capas idénticas, completamente conectadas y basadas en *self-attention*. El método de *self-attention* de esta parte consiste en procesar una oración reemplazando cada elemento por un promedio ponderado de los elementos del resto de la oración.

Cabe mencionar que como *self-attention* es dependiente del orden de la entrada, Transformers le proporciona tanto al *encoder* como al *decoder* vectores que ayudan a determinar la posición de cada palabra (*positional encoding*).

Cada capa del *encoder* (a la izquierda de la figura 1) está constituida por dos subcapas. La primera implementa un mecanismo de *multi-head self-attention*, donde *multi-head* quiere decir que procesa en paralelo distintas funciones de atención sobre la entrada. La segunda subcapa es una red *feed-forward* (las conexiones no forman ciclos). Además, a la salida de cada subcapa del encoder se le suma la entrada de la misma subcapa (conexión residual), y se le aplica una función de normalización (*Layer Normalization*).

El *decoder* (a la derecha de la figura 1) funciona de manera similar, pero además de las dos subcapas de cada capa del *encoder*, se le agrega una tercera subcapa que implementa un mecanismo de atención *multi-head* a la salida del stack del *encoder*. En este caso se aplica un mecanismo de atención *"encoder-decoder"*, que determina interdependencia entre la salida de la primera subcapa del *decoder* y la salida del *encoder* (a diferencia de *self-attention*, que es entre elementos de la entrada).

A su vez, la primera subcapa de atención en el decodificador es una capa de *self-attention* enmascarada, en donde solo se toma atención con respecto a las posiciones anteriores en

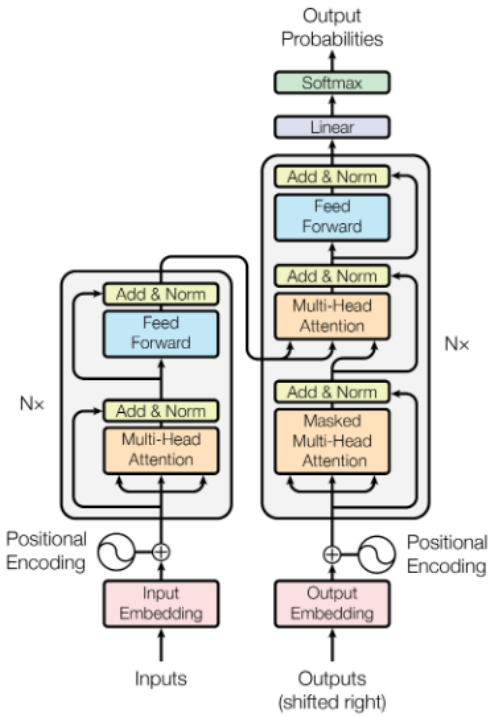

Figura 1: Esquema de la arquitectura del modelo de Transformers. De *Attention is All You Need* [49]

la entrada. Esto se obtiene enmascarando las posiciones futuras antes de hacer el cálculo de *self-attention*.

T5

Text-To-Text Transfer Transformer (T5) es presentado por Raffel et al. [42] como un *framework* unificado que trata todos los distintos problemas del lenguaje basados en texto, como un formato *text-to-text*. Esto es, que todas las tareas consideradas (traducción, clasificación de textos, *question answering*, etc) son manejadas pasándole al modelo un texto de entrada, y entrenándolo para que genere un texto de salida. Esto es en contraste con modelos tipo BERT[14] que solo toman como salida una etiqueta de clase o un *span* del texto de entrada. Al estandarizar las tareas de esta manera se abre la posibilidad que las distintas tareas de PLN utilicen el mismo modelo, función de pérdida e hiperparámetros.

T5 se modela de forma similar al modelo original de Transformers [49], utilizando un esquema de *encoder-decoder* muy parecido. Las tres diferencias que presenta T5 en comparación con Transformers son:

- El uso de una versión simplificada de *layer normalization* que no agrega un sesgo o *bias*.
- La ejecución de la etapa de *layer normalization* antes de la conexión residual, es decir, no se aplica la normalización a la suma de la salida con la entrada de la subcapa, sino solo a la salida.

- El uso de *position embeddings* relativos, en lugar de fijos (Transformers utiliza una función sinusoidal).

GPT-2

GPT-2 es un modelo de lenguaje basado en transformers presentado por Radford et al [41]. Tal como T5, es un modelo muy utilizado para distintas tareas de PLN, como resúmenes automáticos, traducción automática y *question answering*.

GPT-2 funciona generando un token de la salida por vez. Además, es un modelo auto regresivo, es decir, cuando un token es generado por el modelo, se agrega a la entrada, y se utiliza para generar los siguientes tokens.

La principal diferencia que tiene con T5 es que no utiliza *encoder*, sino que solamente utiliza bloques de *decoder*. Estos bloques de *decoder* son como los del modelo de Transformers original, pero no tiene la subcapa de atención *encoder-decoder* que recibe la salida del bloque de *encoder*.

2.1.8. Question generation

La generación de preguntas o question generation es una tarea del procesamiento de lenguaje natural que busca generar preguntas en lenguaje natural basadas en distintos tipos de contenido, como por ejemplo textos, tablas o imágenes. En este trabajo nos enfocamos en textos como fuente de información para esta tarea.

Es posible generar distintos tipos de preguntas, tales como: pregunta-respuesta clásica, múltiple opción, verdadero o falso, completar espacio en blanco, entre otras. En este documento nos centramos en las de tipo pregunta-respuesta clásica.

El trabajo existente de generación de preguntas se puede clasificar en dos grandes ramas: basada en reglas, y utilizando redes neuronales.

Basado en reglas

Este enfoque [13][22] plantea generar las preguntas mediante la utilización de reglas sintácticas y semánticas sobre la fuente de información de entrada. Estas reglas son diseñadas manualmente, lo que es una tarea laboriosa y escala pobremente (cada nuevo tipo de preguntas que se desee incorporar implica diseñar nuevas reglas).

Un ejemplo de este tipo de modelo es el que se utilizó en el proyecto de grado de 2019[34]. En él, se diseñó un conjunto de reglas para cada tipo de pregunta.

Utilizando redes neuronales

En este caso se entrena un modelo neuronal para que reciba la información de entrada y genere las preguntas automáticamente. Estos modelos son de tipo secuencia a secuencia (*seq2seq*) que toman como entrada un texto, que puede ser por ejemplo el contexto que

contiene la respuesta deseada, y a partir de él se produce otro texto, que es la pregunta generada. Este enfoque es mucho más escalable que el basado en reglas, dado que para volúmenes masivos de datos, para lograr un conjunto de reglas apropiado se debería confeccionar una enorme cantidad de reglas. Este enfoque suele obtener mejores resultados que los basados en reglas [51].

En la sección 2.2.5 se muestran ejemplos de trabajos que utilizan este tipo de enfoque.

2.1.9. Métricas

Para evaluar el resultado final obtenido, es necesario hacer uso de alguna medida que permita dar una idea de qué tan bueno es el desempeño logrado. Para ello existen diversas métricas diseñadas para este propósito. Cabe mencionar que hay distintas métricas para cada tarea distinta de procesamiento de lenguaje natural, es decir, las métricas utilizadas para evaluar la generación de preguntas pueden no ser las mismas que se usan para la extracción de información de un texto.

A continuación se detallan algunas de las métricas más comunes y que se utilizaran en este trabajo:

Accuracy

Se utiliza para medir la cercanía de los valores generados frente a valores de referencia.

En el caso de una clasificación binaria, como puede ser ver si los valores generados están o no en el conjunto de referencia, esta medida se puede calcular de la siguiente manera:

$$\frac{VP + VN}{VP + FP + VN + FN}$$

Donde:

- VP = Verdaderos Positivos
- VN = Verdaderos Negativos
- FP = Falsos Positivos
- FN = Falsos negativos

Precision

Esta medida se utiliza para observar qué proporción de los valores generados son relevantes. Se puede ver como una medida que indica la proporción de valores "correctos" de entre los generados. Esta medida puede ser calculada de la siguiente manera:

$$\frac{VP}{VP + FP}$$

Recall

Recall se utiliza para ver qué tantos de los valores del conjunto de referencia fueron realmente generados. Se puede ver como una medida de qué porcentaje del conjunto de referencia se logró cubrir. El cálculo de esta medida es el siguiente:

$$\frac{VP}{VP + FN}$$

F1

Esta métrica[44] se calcula como la media armónica entre la Precision y el Recall:

$$2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} = \frac{VP}{VP + \frac{1}{2}(FP + FN)}$$

Se comporta de manera similar a la Accuracy para problemas con clases balanceadas, pero es más robusta y confiable en problemas desbalanceados, ya que la media armónica es menos sensible a outliers que la media aritmética (la Accuracy se puede pensar como la media aritmética de un conjunto con $(VP + VN)$ elementos que valen 1 y $(FP + FN)$ elementos que valen 0).

Cabe destacar que esta definición de F1 da igual importancia a la Precision y al Recall, mientras que la importancia relativa entre estos dos valores depende del problema. Por ello, también existe una versión ponderada de esta métrica:

$$F_\beta = (1 + \beta^2) \cdot \frac{Precision \cdot Recall}{(\beta^2 \cdot Precision) + Recall}$$

Siendo β un real no negativo que indica cuál de las dos medidas es más considerada. Si β está entre 0 y 1, la medida da mayor importancia al valor de Precision (por ejemplo, $F_0 =$ Precision) mientras que si β es mayor a uno se le da más importancia al valor de Recall. Además, si $\beta = 1$ entonces se obtiene el valor normal no ponderado de F_1.

BLEU

BLEU (Bi-Lingual Evaluation Understudy) es una métrica que compara las preguntas generadas (muchas veces denominada candidata) con las preguntas que se tienen de referencia.[36]

BLEU es una medida diseñada principalmente para la tarea de traducción automática que compara un conjunto de salidas de un sistema de traducción (candidatos) con un conjunto de traducciones consideradas correctas (referencias) y devuelve un valor entre 0 y 1 que indica qué tan bien se pudo cubrir el conjunto de referencia. Esta cualidad de evaluación de similitudes entre conjuntos de secuencias se aprovecha en otras tareas de PLN, como puede ser la generación de resúmenes o la generación de preguntas.

BLEU se calcula a nivel de frases y busca hallar coincidencias entre los n-gramas de la frase candidata y la frase de referencia. Un n-grama es una secuencia de n palabras consecutivas de una oración dada.

Existen distintas variantes de BLEU dependiendo de hasta qué valor de n se tome para la comparación de n-gramas (BLEU-1, BLEU-2, BLEU-3, BLEU-4).

Esta métrica es de alguna manera similar a la de Precision, dado que BLEU básicamente fija qué proporción de los n-gramas de la frase candidata se encuentran en la de referencia. Sin embargo, computar BLEU de la misma manera que Precision conlleva algunos problemas, como el ejemplo de la Tabla 2:

Candidata	el	el	el	el	el	el
Referencia	El	gato	está	en	el	sofá

Tabla 2: Ejemplo 1 BLEU

En este caso, si se calcula el cociente entre los unigramas del candidato que están presentes en la referencia y la cantidad total de unigramas del candidato, el resultado es $6/6 = 1$. Pero como se puede observar, la frase candidata en realidad no es una buena aproximación a la referencia.

Para evitar este caso, BLEU cuenta la cantidad de cada n grama de la frase de referencia y trunca la cantidad de n gramas de la frase candidata si son mayores a estos números. Haciendo esto, en este mismo ejemplo, la frase candidata obtiene una precisión de $2/6$.

Otro problema posible es el de aceptar frases candidatas más cortas que la de referencia, mostrado en la Tabla 3. En este caso la frase candidata podría obtener un buen resultado pero no ser similar a la referencia.

Candidata	el	el				
Referencia	El	gato	está	en	el	sofá

Tabla 3: Ejemplo 2 BLEU

En este caso la frase candidata obtiene un puntaje de $2/2 = 1$, y nuevamente como se ve no es una buena aproximación a la referencia. Para solucionar casos como este, BLEU introduce una penalización por brevedad, que es mayor mientras más chica sea la frase candidata en comparación a la de referencia.

Para el cálculo de BLEU, se calcula individualmente el valor de Precision de cada frase con su referencia correspondiente, considerando el truncamiento planteado en el primer ejemplo. Luego, se calcula la media geométrica de estos valores, y se multiplica por un factor de penalización por brevedad.

La ecuación final es la siguiente:

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

Donde:

- BP es la función de penalización por brevedad, definida como:

$$BP = \begin{cases} 1, & \text{si } c > r. \\ exp(1 - \frac{r}{c}), & \text{si } c < r \end{cases}$$

  siendo c el largo total sumado de las frases candidatas y r el largo total sumado de las frases de referencia.

- w_n son los pesos, que entre todos suman 1.

- p_n es la media geométrica de los valores de Precision para los n-gramas.

En la expresión del cálculo de BLEU-N 1, el extremo N de la sumatoria indica que variante de BLEU se está utilizando: para determinado valor "N" se estará usando la variante BLEU-N, con $N \in \{1, 2, 3, 4\}$.

METEOR

METEOR (Metric for Evaluation of Translation with Explicit ORdering) es una métrica para la evaluación de traducción automática. Se basa en una generalización del concepto de matching de unigramas entre texto generado por computadores y referencias producidas por humanos.[27]

METEOR evalúa las hipótesis mediante una alineación con su referencia y calculando puntajes de similaridad a nivel de oración.

Para un par hipótesis-referencia, el espacio de alineamientos se genera revisando todas las posibilidades de emparejamiento entre las oraciones, de acuerdo a los siguientes emparejadores:

Exacto Dos palabras se emparejan si son idénticas.

Raíz Dos palabras se emparejan si tienen la misma palabra raíz, de acuerdo a Snowball Stemmer.

Sinónimo Dos palabras se emparejan si pertenecen al mismo conjunto de sinónimos según la base de datos de WordNet.

Paráfrasis Dos palabras se emparejan si son listadas como paráfrasis en una tabla de paráfrasis acorde al lenguaje tratado (en este caso, Inglés).

Una vez un par hipótesis-referencia está alineado, se identifican palabras léxicas y gramaticales (en inglés content y function words), dentro de la hipótesis (h_c y h_f, respectivamente) y de la referencia (r_c y r_f, respectivamente). Una palabra gramatical se define como una palabra que ocurre con una frecuencia mayor a 10^-3 dentro del conjunto de referencia.

Luego, para cada uno de los emparejadores (m_i) se cuenta la cantidad de emparejamientos para las palabras de contenido y función de la hipótesis ($m_i(h_c)$ y $m_i(h_f)$) y de la referencia ($m_i(r_c)$ y $m_i(r_f)$).

Se calculan valores de Precision y Recall ponderados utilizando distintos pesos para cada emparejador (w_i) y una constante de ponderación para las palabras de función (δ):

$$P = \frac{\sum_i w_i \cdot (\delta m_i \cdot h_c + (1-\delta) m_t \cdot h_f)}{\delta \cdot \mid h_c \mid + (1-\delta) \cdot \mid h_f \mid}$$

$$R = \frac{\sum_i w_i \cdot (\delta m_i \cdot r_c + (1-\delta) m_t \cdot r_f)}{\delta \cdot \mid r_c \mid + (1-\delta) \cdot \mid r_f \mid}$$

Ahora, se calcula la media harmonica parametrizada de P y R:

$$F_{mean} = \frac{P \cdot R}{\alpha \cdot P + (1-\alpha) \cdot R}$$

Se agrega una penalización sobre espacios y diferencias en el orden de las palabras. Esta es calculada usando el número total de palabras emparejadas, tanto para la hipótesis como para la referencia (m) y el número de chunks (ch), donde un "chunk" se define como una serie de emparejamientos contiguos y ordenados de la misma manera en la hipótesis y la referencia. El cálculo de la penalización es el siguiente:

$$Pen = \gamma \cdot \left(\frac{ch}{m}\right)^{\beta}$$

Luego, el puntaje final de METEOR se calcula de la siguiente manera:

$$METEOR = (1 - Pen) \cdot F_{mean}$$

ROUGE

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) es un conjunto de métricas usado para evaluar resúmenes automáticos y traducción automática. Comparan un texto generado automáticamente con uno de referencia. El hecho de que sea recall-oriented significa que la métrica simboliza cuánto de el texto de referencia está en el texto generado.[29]

En particular ROUGE-L mide las secuencias más largas de coincidencias usando Longest Common Subsequence (LCS).

2.2. Trabajos relacionados

2.2.1. SQuAD: 100,000+ Questions for Machine Comprehension of Text

Rajpurkar et al. (2016) presentó el Stanford Question Answering Dataset (SQuAD), un corpus ampliamente utilizado en el problema de generación de preguntas y respuestas [43].

Este dataset consiste en más de 100.000 preguntas extraídas de artículos de Wikipedia, en donde la respuesta a la pregunta se puede encontrar en el texto correspondiente.

Para su creación se eligieron al azar 536 artículos de entre los mejores 10.000 artículos en idioma inglés de Wikipedia, según el algoritmo PageRanks.

Luego, se extrajeron párrafos individuales, descartando imágenes, figuras y tablas y excluyendo aquellos artículos que contuvieran más de 500 palabras.

Como resultado se obtuvieron 23.215 párrafos de los 536 artículos iniciales.

Para la generación de pares de preguntas y respuestas, se emplearon anotadores pagos a través de la plataforma Daemo. A cada trabajador se le pidió que generase 5 pares pregunta-respuesta por cada párrafo que tuviera. Como requerimiento se pidió que las preguntas se pudieran responder con un span del texto original.

Por último, se obtuvieron al menos dos respuestas más para cada pregunta, para generar una evaluación más robusta y obtener un indicador del desempeño humano en SQuAD. Esto se realizó mostrándole una pregunta y el texto del cual se obtuvo la misma a un trabajador y pidiéndole que marque en el texto el span donde se encuentra la respuesta. En caso de que la pregunta no pudiera ser respondida por un span del texto, se pidió que los anotadores dejaran la respuesta como vacía. Como resultado, 2.6 % de las preguntas son marcadas como imposibles de responder por un span de texto.

2.2.2. NewsQA: A machine comprehension dataset

Otro corpus bastante utilizado es el de NewsQA (Trischler et al. 2017)[48]. Este conjunto de datos consiste en más de 100.000 pares pregunta-respuesta generados por humanos basados en un conjunto de alrededor de 10.000 artículos de la CNN.

Similar a lo que se hace en SQuAD, se obtuvieron artículos de CNN usando un script. De los 90.266 artículos obtenidos se eligieron aleatoriamente 12.744.

Luego, se emplearon anotadores pagos para generar las preguntas. Se les proveyó solamente del titular y un pequeño resumen de la noticia y se les pidió que en base a ello generaran hasta 3 preguntas. No se les proveyó del artículo original con el fin de no obtener preguntas que fueran reformulaciones de las palabras del texto. Esto también generó la posibilidad de que una pregunta no pueda ser respondida solo con el contenido del artículo. También, se rechazaron preguntas que coincidan fuertemente con el resumen dado.

Un segundo conjunto de anotadores se encargó de responder las preguntas generadas por el primer conjunto. Estos recibieron el artículo completo y las preguntas generadas para ese artículo. El trabajador podía responder la pregunta, marcando las palabras que las responden dentro del texto (se les indicó que si es posible, marquen un span continuo), marcar la pregunta como sin sentido o seleccionar una respuesta “nula” si la respuesta no se encontraba en el texto.

Para cada pregunta se solicitó la respuesta a varios anotadores, con el objetivo de alcanzar un acuerdo en al menos dos de ellos.

Para las preguntas que no cumplieron con esta última condición (no se alcance acuerdo entre dos o más anotadores), se realizó una etapa de validación. En ella un tercer conjunto

de anotadores recibió el artículo completo, las preguntas de dicho artículo y todas las respuestas generadas para cada una de estas preguntas. El trabajador debía marcar la respuesta que le pareciera que mejor respondía la pregunta, o rechazar todas las respuestas.

Luego de la etapa de validación, un 86 % de las preguntas tuvieron respuestas que pudieron ser acordadas por al menos dos anotadores. Se decidió que las preguntas restantes también fueran incluidas en el conjunto de datos final, pero estas están específicamente marcadas.

Por último, se combinaron spans de la misma respuesta que estén a menos de 3 palabras de distancia.

2.2.3. Learning to Ask: Neural Question Generation for Reading Comprehension

Du et al. (2017)[17] presentó este trabajo acerca de generación de preguntas utilizando redes neuronales, que es uno muy referenciado por trabajos posteriores.

En él se planteó un modelo de lenguaje, donde la probabilidad condicional de cada palabra dada las anteriores se modela usando una arquitectura de codificador-decodificador RNN [8][11]. Se adoptó un mecanismo de atención global[31] para que el modelo se concentre en ciertos elementos de la entrada al decodificar. Se plantean dos alternativas del modelo, uno que codifica a nivel de oración y otro que codifica tanto a nivel de oración como de párrafo.

Además de los avances que plantea este artículo, se deja disponible la división del corpus de SQuAD utilizada en el mismo. Para preparar el dataset se realizaron los siguientes pasos:

- Se extrajeron oraciones y se emparejaron con las preguntas
- Se corrió Stanford CoreNLP para la tokenización y el split de las oraciones
- Se pasó todo el dataset a minúscula
- Con el offset de las respuestas a cada pregunta se identificó la oración que contiene la respuesta y se utiliza esta oración como entrada. En algunos casos se observó que la respuesta estaba comprendida entre dos oraciones, en esta situación se tomó como entrada la concatenación de ambas oraciones
- Se agregó la etiqueta <SOS> al principio y la etiqueta <EOS> al final de cada oración

Por último, se dividió el dataset de manera aleatoria en un conjunto de entrenamiento (80 %), un conjunto de evaluación (10 %) y un conjunto de prueba (10 %).

Este dataset con esta misma división se encuentra disponible y es utilizado por muchos trabajos, por lo que su uso tiene la ventaja de presentar muchos puntos de comparación posibles.

2.2.4. Construcción de herramientas para enseñanza de inglés: generación de preguntas y respuestas - Proyecto de Grado 2019

Nuestro trabajo surge luego de la realización de este proyecto de grado en 2019[34], que a su vez se construye sobre la base de otros proyectos previos realizados en 2018. Este proyecto se planteó con el mismo objetivo que el actual: generar herramientas para la enseñanza de inglés a nivel básico, con el fin de ser usadas por docentes para enseñar dicho idioma a nivel escolar.

La diferencia con nuestro trabajo radica en el enfoque utilizado para la generación de las preguntas y las respuestas. En contraste al enfoque de redes neuronales utilizando modelos de lenguaje utilizado en este trabajo, en el proyecto de 2019 se usó un enfoque basado en reglas, en el cual a partir del texto de entrada y un conjunto de reglas se obtiene un conjunto de preguntas y respuestas sobre dicho texto.

En este caso el proceso o pipeline de generación de preguntas y respuestas viene dado por las siguientes etapas:

Preprocesamiento del texto de la entrada

En esta etapa se realiza un preprocesamiento del texto de manera que luego el conjunto de reglas diseñado disponga de la suficiente información para funcionar correctamente.

Hay cuatro componentes que son ejecutados en esta etapa:

- POS Tagging
- Semantic Role Labeling
- Coreference Resolution
- Named Entity Recognition

Cada uno de estos componentes es implementado utilizando la funcionalidad correspondiente de la librería AllenNLP[18].

Ejecución de reglas

Las reglas diseñadas requieren del etiquetado de roles semánticos (Semantic Role Labeling), el etiquetado gramatical (POS Tagging), la resolución de correferencias (Coreference Resolution) y la identificación de entidades con nombre dentro el texto (Named Entity Recognition). Toda esta información se obtiene del preprocesamiento del paso anterior.

Se confecciona un conjunto de reglas para cada tipo de pregunta para los tipos de pregunta más típicas: *Who, What, Where y When*. Además, se agrega una regla para el tipo de pregunta *What colour*. Esta decisión surge de observar que la regla para las preguntas de tipo *What* no era adecuada para las preguntas de tipo *What colour* y estas últimas, en el

contexto del nivel de inglés sobre el que se trabajó, eran lo suficientemente comunes como para tener su propia regla.

Clasificación o ranking de preguntas

Una problemática que surgió fue la generación de una indeterminada (y eventualmente muy grande) cantidad de preguntas, lo que hace que, si bien los docentes pueden explorar las preguntas generadas y seleccionar aquellas que deseen conservar, deban recorrer una gran lista de candidatos. Para solucionar esto, se resolvió ordenar las preguntas generadas mediante distintos criterios y seleccionar y desplegar al usuario solamente las primeras cinco (cantidad modificable) de este *ranking*.

Para este ranking se utilizan dos criterios:

- **La posición dentro del texto de entrada de la oración que generó la pregunta** con el fin de que las preguntas generadas traten sobre distintas partes del texto y no se solamente en una sección.
- **La cantidad de preguntas de generadas de cada tipo** para así priorizar aquellas que aparezcan en menor cantidad, en un intento de aumentar la variedad de las preguntas extraídas.

2.2.5. Otros trabajos

A continuación se mencionan otros trabajos de generación de preguntas utilizando redes neuronales. En la sección 5.1 se muestran los resultados obtenidos en cada uno de ellos y una comparación con los resultados obtenidos por nuestros modelos.

Para la generación de preguntas, un esquema muy comúnmente utilizado es el de *encoder-decoder* con atención. Du et al. [17] fue de los primeros trabajos que utilizaron este esquema, y utilizaron dos variaciones para el modelo: tomando información a nivel de oración y tomando información a nivel de párrafo.

Zhou et al. [52] también utilizan un modelo *encoder-decoder* basado en atención. Toman como entrada la oración, información de la posición de la respuesta y dos rasgos lingüísticos (*POS* y *NER*). Además utilizan un mecanismo de copia [21][20] para copiar directamente palabras poco comunes de la oración original.

Du and Cardie [16] propusieron adjuntar a cada pronombre de la entrada, el antecedente más "representativo" que surgiera de una resolución de correferencias. También se mantiene información posicional de cada una de estas parejas pronombre-antecedente.

Song et al. [47] utiliza un *matching* entre la respuesta y el pasaje para obtener información contextual relevante, o, dicho de otra forma, detecta qué palabras del pasaje son relevantes en el contexto de la pregunta.

Liu et al. [30] introduce un *clue word predictor*, que predice palabras potenciales que pueden aparecer en la pregunta objetivo. Incorpora esto junto con otras *features* como

características léxicas e indicadores de posición de la respuesta.

Dong et al. [15] presenta el *Unified Pre-trained Language Mode*l (*UNILM*), que es un modelo de lenguaje grande, pre-entrenado, basado en *transformers* que utiliza atención y puede ser finetuneado para tareas de generación de preguntas. Proponen tomar como entrada un pasaje del texto y el *span* de la respuesta, y generar como salida una pregunta para esa respuesta.

3. Corpus

Los corpus son un elemento crucial para este trabajo, dado que son la fuente principal de información que alimenta los modelos neuronales para que estos puedan ser entrenados correctamente, así como una herramienta importante en su evaluación.

En esta sección se presentan los corpus utilizados en el transcurso del trabajo, tanto para el entrenamiento como para la evaluación de los modelos. Primero, se muestran corpus existentes y de uso popular en trabajos relacionados y que permiten comparar los resultados obtenidos con los de estos trabajos. Luego, se presentan los corpus generados para este trabajo así como uno existente generado en el Proyecto de Grado 2019[34], que fue adaptado y corregido para formar un corpus en conjunto con el generado en este trabajo. Estos se desarrollaron con el objetivo del trabajo en mente por lo que son más apropiados para la evaluación de los modelos que los demás corpus.

3.1. Corpus generales para QA

En este trabajo se utilizan corpus de preguntas y respuestas sobre textos. Estos contienen, al menos:

- Textos sobre los que se extraen preguntas y respuestas
- Preguntas extraídas de dichos textos
- Respuestas extraídas de dichos textos

El principal uso de corpus como el SQuAD o el NewsQA es el de generación de respuestas, que es una tarea similar a la buscada, pero en la que ya se dispone de la pregunta que necesita ser respondida.

Por lo general los corpus de preguntas y respuestas son generados y utilizados para la tarea de *question answering*, en donde se toma como entrada el contexto y la pregunta para generar la respuesta, y comparar los resultados contra las respuestas del corpus. En nuestro caso usamos los corpus de una forma diferente. Como los corpus incluyen tanto los contextos, como las preguntas y respuestas, podemos utilizar como entrada a nuestro modelo el texto y la respuesta, y a partir de eso generar las preguntas, pudiendo evaluar nuestro rendimiento contra las preguntas de los corpus. Nuestro enfoque es similar al de varios de los trabajos que presentamos en la sección 2.1.8.

Se decidió utilizar estos dos corpus ya que:

- **Son corpus grandes, ambos de 100.000+ preguntas:** Esto es necesario puesto que van a ser utilizados para entrenar modelos neuronales.
- **Son de uso común en muchos trabajos previos:** Se ofrecen muchos puntos de comparación, en particular SQuAD es usado por muchos grupos como *benchmark* para evaluar los modelos de generación de preguntas.
- **Todas sus preguntas se pueden responder con un *span* específico dentro del texto correspondiente:** Se busca generar preguntas en las que la respuesta se

encuentre directamente en el texto, en vez de buscar respuestas más complejas que no necesariamente aparezcan en el texto.

- **Textos tipo narraciones o artículos en vez de conversaciones:** Estos tipos de texto se asemejan más a los que se pueden encontrar en evaluaciones de inglés.

Los textos de SQuAD son sacados de artículos de Wikipedia, mientras que los de NewsQA son extraídos de artículos de CNN. En ambos casos son textos con un nivel de inglés superior al deseado, además de que no están pensados para un uso educacional. Por esta razón los utilizamos solamente en el entrenamiento de los modelos, y no para su evaluación.

En las tablas 4 y 5 se muestra cómo se distribuyen los tipos de pregunta dentro de cada corpus. Los datos fueron tomados contando la cantidad de veces que aparece cada palabra al inicio de una pregunta, y seleccionando en particular *"what"*, *"who"*, *"when"*, *"where"* y *"how"* que son las que buscamos generar, y además resultan ser de las más comunes en ambos *datasets*. También se muestra el porcentaje para cada caso (este porcentaje no suma 100 % por la existencia de preguntas que empiezan con otras palabras).

SQuAD	*What*	*Who*	*When*	*Where*	*How*	Total
dev	8359 45,9 %	1649 9 %	1082 5,9 %	835 4,6 %	1892 10,4 %	18216 100 %
test	4783 40,3 %	1227 10,3 %	637 5,4 %	409 3,4 %	1206 10,2 %	11877 100 %
train	32810 43,3 %	6923 9,1 %	4822 6,4 %	2882 3,8 %	6918 9,1 %	75722 100 %

Tabla 4: Distribución por tipo de pregunta de SQuAD

NewsQA	*What*	*Who*	*When*	*Where*	*How*	Total
dev	2599 43,4 %	1196 20 %	233 3,9 %	457 7,6 %	394 6,6 %	5988 100 %
test	2602 43,6 %	1186 19,9 %	214 3,6 %	427 7,2 %	441 7,4 %	5971 100 %
train	47448 44,1 %	20620 19,2 %	4465 19,2 %	7616 4,1 %	7352 6,8 %	107672 100 %

Tabla 5: Distribución por tipo de pregunta de NewsQA

Como se puede ver, en ambos corpus hay una predominancia importante de las preguntas de tipo *what*, que en ambos casos superan el 40 % de las preguntas totales. En el caso de SQuAD, estas son seguidas por las de tipo *who* y *how*, con una cantidad casi igual de preguntas.

Por otro lado, en NewsQA también las preguntas de tipo *who* y *how* son las segundas y terceras, respectivamente, pero a diferencia de SQuAD, hay una cantidad considerablemente más grande de preguntas de tipo *who* sobre las de tipo *how*.

Por último se tienen las preguntas de tipo *when* y *where*, que son las que aparecen una menor cantidad de veces. En SQuAD hay más preguntas de tipo *when* mientras que en NewsQA es al revés, se tienen más preguntas de tipo *where*.

3.2. Corpus específicos para generación de preguntas de nivel inicial

Para la evaluación de los modelos entrenados, se quiso disponer de un corpus de preguntas y respuestas extraídas de textos de inglés de un nivel similar al que se pretende enseñar utilizando la herramienta desarrollada.

Debido a esto, se propuso crear un corpus del nivel de inglés que se busca evaluar. Para ello, se buscaron textos de niveles de inglés A1 y A2, que son los niveles más básicos y son los que se busca enseñar a estudiantes que recién comienzan con su estudio de este idioma.

Este corpus fue diseñado en base a dichos textos, extraídos de la web [4], donde las preguntas y respuestas fueron confeccionadas manualmente y siempre considerando que la respuesta sea un fragmento del texto correspondiente. Además, se agrega el fragmento de texto donde se puede encontrar la respuesta con tal de simplificar algunas operaciones posteriores. Este corpus contiene 5 textos y 138 pares pregunta-respuesta.

Para la confección de los pares pregunta-respuesta presentes en este corpus se fue viendo oración por oración, para cada texto, todas las preguntas que se podían realizar, siempre y cuando fueran coherentes y relevantes para al texto. Estos pares eran luego anotados junto con el texto de donde fueron extraídos así como dos números que indican el primer y último carácter del span del texto que contiene la respuesta.

El corpus fue guardado en formato CSV (Comma Separated Values), donde cada línea es de la siguiente forma:

$$Texto, Pregunta, Respuesta, InicioSpan, FinSpan$$

Dos formas comunes de almacenar esta información son denominadas AQPL (All Questions Per Line) y OQPL (One Question Per Line).

La primera almacena un texto y a continuación todos sus pares de preguntas y respuestas vinculados separados por un delimitador seleccionado. Este formato evita la redundancia de almacenar cada texto una vez por cada pregunta-respuesta vinculada pero agrega dificultad a la hora de recuperar la información.

El segundo formato, OQPL, guarda un texto seguido de uno de sus pares de pregunta-respuesta, haciendo que el mismo se repita tantas veces como pares vinculados tenga. Esto facilita la recuperación de la información pero introduce una redundancia que hace que el corpus final tenga un tamaño mayor y por consiguiente, hace que la etapa de entrenamiento tenga una duración mayor.

En este trabajo se optó por el segundo formato pues simplifica el uso posterior del corpus y al ser un conjunto de datos de tamaño moderado, dicha redundancia no genera un gran impacto.

Además de este corpus, construido por los autores del presente proyecto (que denominaremos corpus 2021), se dispuso de un corpus existente construido por los autores del

Proyecto de Grado 2019[34]. Este corpus fue confeccionado en su momento de manera similar al mencionado anteriormente y también presenta el nivel de inglés deseado. Sin embargo, fue necesario adaptarlo para ajustarlo a las necesidades de este proyecto. Dicha adaptación consistió en llevar este corpus al formato mencionado anteriormente, así como algunas correcciones de los pares pregunta-respuesta que contenía. Este corpus está compuesto por 20 textos y 316 pares pregunta-respuesta.

Se decidió fusionar el corpus de 2019 con el de 2021 para la etapa de evaluación. Este corpus combinado contiene 25 textos y 454 pares pregunta-respuesta.

En la tabla 3.2 y en la figura 2 se muestran la cantidad de preguntas de cada tipo en cada uno de los conjuntos de datos mencionados.

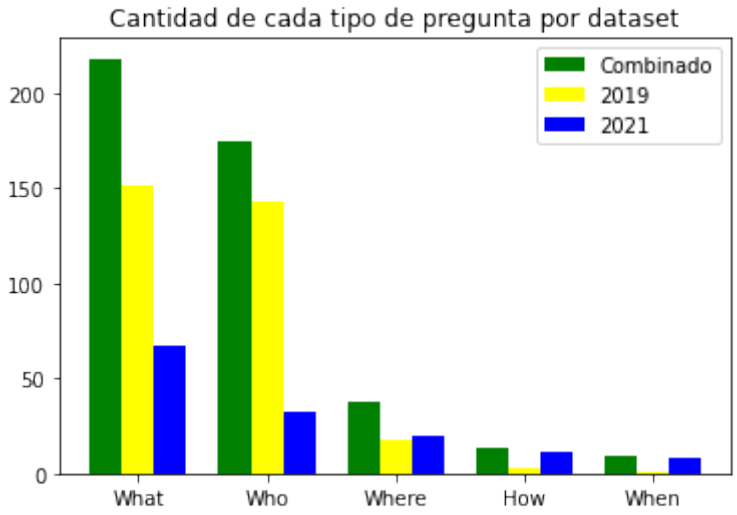

Figura 2: Distribución de tipos de pregunta de los datasets 2019, 2021 y el fusionado

Corpus	*Textos*	*What*	*Who*	*When*	*Where*	*How*	Total
2021	5	75	24	6	20	13	138
		54,3 %	17,4 %	4,3 %	14,5 %	9,4 %	100 %
2019	20	143	151	3	18	1	316
		45,3 %	47,8 %	0,9 %	5,/ %	0,3 %	100 %
2021+2019	25	218	175	9	38	14	454
		48 %	38,5 %	2 %	8,4 %	3,1 %	100 %

Tabla 6: Distribución de tipos de pregunta de los datasets 2019, 2021 y el fusionado

Como se puede observar, la mayor parte de los corpus se concentra en preguntas de *Who* y *What*, lo que se debe a que en los textos que constituyen el dataset, suelen describirse entidades realizando acciones, de donde se sacan las preguntas de *Who* y *What*, mientras que no hay demasiadas referencias temporales, de lugar o de modo, por lo que tiene sentido que las preguntas de tipo *When*, *Where* y *How* sean menos frecuentes.

Otro aspecto a notar acerca de la tabla es que, mientras que en el proyecto de 2019 se diferenció entre las preguntas de tipo *what* y las de tipo *what colour*, en este trabajo se decidió no hacer esta distinción, y entonces todas las preguntas que empiezan con *what color* se incluyen dentro de la columna de *what*.

Además, cabe destacar que al agregar el dataset de 2021 al de 2019, el tipo predominante

de pregunta pasa de ser *Who* (aunque por muy poco margen) al tipo *What* (si bien el tipo *Who* sigue siendo muy significativo).

Por último, se puede ver que en el dataset de 2019 solamente hay una pregunta de tipo *How* y tres de tipo *When*, pero al fusionar estos dos datasets, se logra aumentar la cantidad de preguntas de estos tipos a 14 y 9, respectivamente.

A modo de resumen, se presenta la Tabla 7 que indica para qué fue utilizado cada corpus.

	Entrenar modelos neuronales	Evaluar generación de preguntas	Evaluar selección de respuestas	Evaluar pipeline completo
NewsQA train	✓			
SQuAD train	✓			
NewsQA y SQuAD train	✓			
NewsQA test		✓	✓	
SQuAD test		✓	✓	
2019+2021		✓	✓	✓

Tabla 7: Etapas en las que se utilizó cada corpus

Como se puede apreciar en la tabla, se utilizaron los conjuntos train de SQuAD y NewsQA, tanto individualmente como en conjunto (utilizando los dos de forma secuencial), para entrenar los modelos. Por otro lado, los conjuntos de test de SQuAD y NewsQA, así como el corpus generado en este trabajo fueron utilizados para la evaluación de las etapas de generación de preguntas de selección de respuestas, de forma separada, pero solo el último fue utilizado para evaluar el proceso completo.

4. Desarrollo

En esta sección se presenta cómo es el proceso de principio a fin de generación de un conjunto de preguntas y respuestas partiendo de un texto plano.

En su forma más básica este proceso consiste de dos etapas: una de selección de respuestas desde el texto y otra de generación de preguntas en base a dichas respuestas. También es posible juntar estas dos etapas en una sola, pero se optó por hacerlo en dos para poder aprovechar el material existente tanto para generación de preguntas como de extracción de información.

Al implementar este proceso con las dos etapas mencionadas, surgieron algunos problemas que ameritaron la introducción de una etapa de preprocesamiento y otra etapa de post procesamiento al comienzo y al final del proceso, respectivamente.

En la Figura 3 se detalla la ejecución del pipeline y cada una de sus etapas. Como se puede ver, el pipeline consiste en cuatro etapas, donde en cada una se utilizan herramientas distintas para generar como salida la entrada de la siguiente etapa.

La etapa de selección de respuestas se resuelve utilizando una librería de semantic role labeling de la herramienta AllenNLP [46] para seleccionar ciertos roles semánticos como respuestas, mientras que la de generación de preguntas se resuelve con un enfoque neuronal utilizando un modelo basado en T5 o GPT-2.

Previo a la selección de respuestas se incluye una etapa de preprocesamiento del texto de entrada, y luego de generar las preguntas se incluye una etapa de postprocesamiento de la salida.

A continuación se muestra un ejemplo de la ejecución del pipeline, mostrando la salida de cada una de las etapas, para el siguiente texto:

Adriana is tall. She has brown hair. Mathew is eleven. He likes basketball.

Al pasar por la etapa de preprocesamiento (ver sección 4.1), se obtiene un texto con sus correferencias resueltas.

Adriana is tall. Adriana has brown hair. Mathew is eleven. Mathew likes basketball.

Luego, en la etapa de selección de preguntas (ver sección 4.2), se obtienen una serie de respuestas candidatas del texto.

[*'Adriana', 'tall', 'Adriana', 'brown hair', 'Mathew', 'eleven', 'Mathew', 'basketball'*]

Cabe mencionar que "Adriana" y "Mathew" aparecen dos veces porque son dos instancias distintas dentro del mismo texto. La salida literal de esta etapa es una serie de copias del texto donde la respuesta candidata está comprendida entre dos etiquetas <hl>, por lo que de esta manera se puede diferenciar cada instancia.

[*' <hl> Adriana <hl> is tall. Adriana has brown hair. Mathew is eleven. Mathew likes basketball.', ... otros textos, ...*]

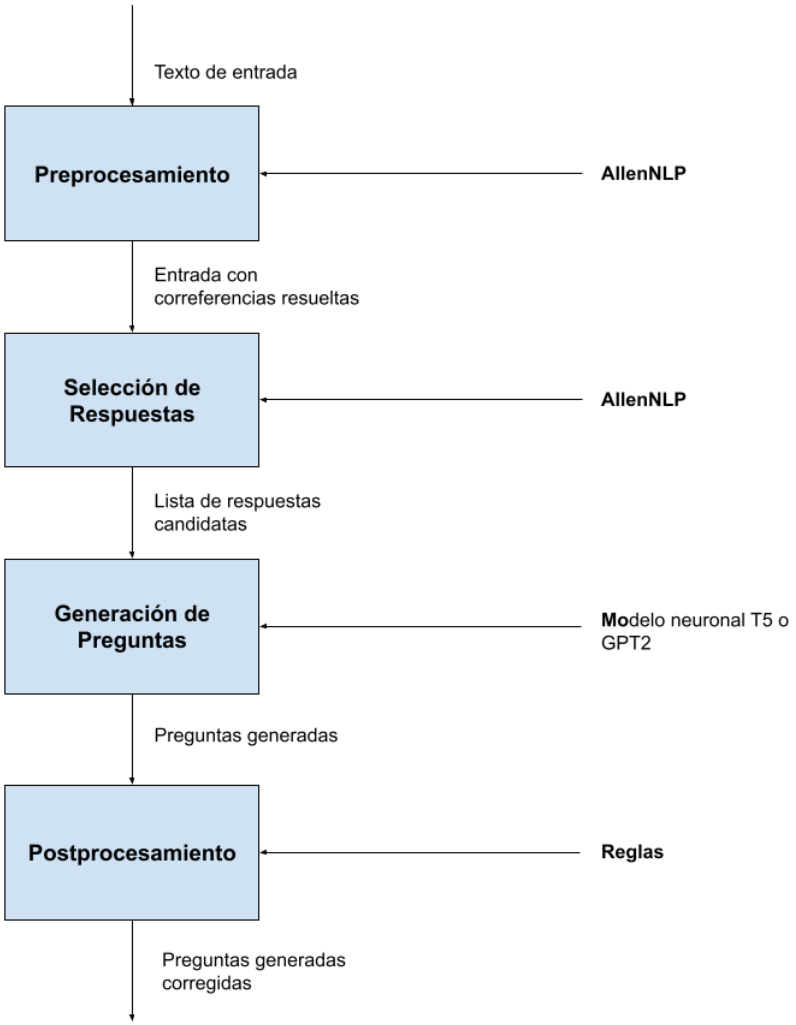

Figura 3: Esquema de ejecución del pipeline

La siguiente etapa es la de generación de preguntas (ver sección 4.3), en donde se generan una o más preguntas para cada una de las respuestas candidatas que se obtienen de la parte anterior. La cantidad de preguntas a generar por cada respuesta es un parámetro del modelo, por defecto se usa 3.

[*'pregunta': 'Who is tall?', 'respuesta': 'Adriana', ... otras preguntas, ...*]

Por último, en la etapa de postprocesamiento (ver sección 4.4), se arreglan posibles malformaciones que se generen en las preguntas.

Antes del postprocesamiento	Luego del postprocesamiento
Who likes basketball??????	Who likes basketball?
Who likes basketball? Who is tall? Who has brown hair?	Who likes basketball?

Se deja una *demo* de la herramienta disponible en Google Colab[1] que permite probar el

[1] https://colab.research.google.com/drive/1LTVKTZVfcijplBJAnyiFY417QpuO8PWe?usp=sharing

pipeline entero con las variantes de los modelos de generación de preguntas y selección de respuestas que obtuvieron mejores resultados en la sección 5.

Estas cuatro etapas de preprocesamiento, selección de respuesta, generación de preguntas y postprocesamiento son las que constituyen el pipeline, y se describen en detalle a continuación.

4.1. Preprocesamiento

En un principio se experimentó con un pipeline de dos etapas, selección de respuestas y generación de preguntas. Sin embargo se observó que muchas veces, si en el texto aparecen pronombres, el par pregunta-respuesta generado podía presentar imprecisiones. Se identificaron dos de estos casos. El primer caso sucede cuando en la pregunta generada aparece un pronombre, lo que introduce un cierto grado de ambigüedad si en el texto hay más de una referencia posible para este pronombre. El segundo caso es cuando la respuesta identificada es un pronombre, lo que hace que la propia respuesta sea ambigua.

Por estas razones, se decidió agregar una etapa de preprocesamiento en la que se aplique un resolvedor de correferencias al texto original, de manera que se evite tomar pronombres como respuestas o que se generen en la pregunta y, en su lugar, tomar la palabra que referencian. Más adelante, en la sección 5, se muestran resultados con y sin esta etapa de resolución de correferencias que ilustra la mejoría que se obtuvo al incluirla.

Se utiliza un modelo de resolución de correferencias sobre todo el texto de entrada. Se usó la librería de AllenNLP, y en particular el modelo de tipo SpanBERT de resolución de correferencias preentrenado[28]. Se utilizó el método coref_resolved del predictor para obtener un texto donde las correferencias se sustituyen por la principal mención. Para todas las etapas siguientes pasamos como entrada el texto con las correferencias sustituidas.

En la tabla 8 se muestran ejemplos de preguntas generadas a partir de la oración original y la oración con las correferencias resueltas. En el primer ejemplo se puede ver cómo se obtiene una respuesta más descriptiva, *Mathew*, en lugar de *He*. En el segundo ejemplo vemos un cambio en la pregunta en lugar de la respuesta, cambiando *her* por *Haley* y *the doctor* por *a doctor*. Ya que la pregunta está separada del texto, por más que con la palabra *her* sepamos a quién se refiere, es más adecuado que aparezca el nombre propio, *Haley*, para dar más claridad a la pregunta. Sin embargo, podemos ver cómo en esta pregunta se cambia *the doctor* por *a doctor*. Esto en principio es un error, ya que es más adecuada la versión con *the*. De esto podemos pensar que la resolución de correferencias también puede introducir errores, lo que abordaremos más abajo.

Otra cosa que podemos observar es que el modelo de generación de preguntas es capaz de corregir algunas de las correferencias por sí mismo correctamente como en el caso de corregir *he* a *the doctor*, pero no otras como en el caso de *her* a *Haley*. Realizar la resolución de correferencias antes de pasarle el texto al modelo generador de preguntas garantiza que siempre se corrijan.

Debemos mencionar que la resolución de correferencias no resulta en una mejora estricta en todos los casos, sino que en algunos casos cambia un pronombre por otro. Por ejemplo,

Texto	Sin resolución correferencias	Resolviendo correferencias
Mathew is eleven. He likes basketball.	**Pregunta**: Who likes basketball? **Respuesta**: *He*	**Pregunta**: Who likes basketball? **Respuesta**: *Mathew*
Haley feels hot. Her mom touches her forehead. Haley has a fever. The mom takes Haley to a doctor. The doctor is kind. He gives her a sticker. He tells her to take a pill.	**Pregunta**: What does *the* doctor tell *her*? **Respuesta**: to take a pill	**Pregunta**: What does *a* doctor tell *Haley*?, **Respuesta**: to take a pill

Tabla 8: Ejemplo de preguntas y respuestas generadas con y sin resolver correferencias

al resolver las correferencias de una oración, se resuelven de a una, como ilustra el siguiente ejemplo:

Tomorrow is the first day of school. Barbara looks at her backpack. It has holes. It is dirty.

Tomorrow is the first day of school. Barbara looks at Barbara's backpack. Her backpack has holes. Her backpack is dirty.

Aquí, la palabra *it* se resuelve como *her backpack*, lo cual es correcto, pero introduce el pronombre *her*. Esto se podría solucionar resolviendo correferencias otra vez, sobre esta nueva oración.

También cabe destacar que la resolución de correferencias puede introducir errores. En el último ejemplo, luego de resolver las correferencias, la pregunta "*What does Barbara look at*?", tendría como respuesta "*Barbara's backpack*", lo que es redundante, en lugar de la respuesta más adecuada, que sería "*Her backpack*".

El siguiente ejemplo ilustra otro ejemplo de error cometido al resolver las correferencias de una oración:

It is Lily's birthday. She gets out of bed. Nobody says anything to her. Her mom takes her to school.

Luego de resolver las correferencias, la oración resultante es la siguiente:

It is Lily's birthday. Lily's gets out of bed. Nobody says anything to Lily's. Lily's's mom takes Lily's to school.

Como la primera instancia en la que se nombra a Lily aparece de forma posesiva, todos los pronombres que refieran a este nombre se reemplazan como posesivos también, por más que no lo fuera en un principio.

Dicho esto, se decidió de igual manera utilizar la resolución de correferencias dado que la cantidad de errores solucionados por esta tarea era mucho mayor que la cantidad de errores introducidos. Esto se muestra de mejor manera al comparar los resultados obtenidos por los distintos modelos con y sin esta tarea incluida, en la sección de evaluación experimental.

4.2. Selección de respuesta

En esta etapa se busca obtener una colección de candidatos a respuesta a partir del texto de la entrada. Para seleccionar estos candidatos, se decidió utilizar *semantic role labeling* para etiquetar las distintas palabras del texto, y luego tomar aquellas cuyo rol se encuentre en una lista de roles determinada.

Para realizar esto, el modelo de SRL utilizado es el *Structured Prediction SRL-BERT* de *AllenNLP*. Se optó por considerar las etiquetas ARG0, ARG1, ARG2, ARG3, ARG4, ARGM-TMP y ARGM-LOC para la generación de candidatos a respuesta. La tabla 9 muestra ejemplos de algunos candidatos a respuesta obtenidos mediante estas etiquetas.

Para la elección de este conjunto se consideraron los roles semánticos que responden preguntas de tipo *What*, *Who*, *When* y *Where*, que son los roles considerados por el proyecto de 2019 y además de los tipos de pregunta más comunes del idioma. A diferencia del proyecto de 2019, en el que se incluyen reglas específicas para preguntas de tipo *What colour*, aquí no hacemos esta distinción, pero igualmente este tipo de preguntas pueden ser generadas. Cabe mencionar que el conjunto de etiquetas considerado no genera solamente preguntas de los tipos mencionados, sino que también genera de otros, como por ejemplo preguntas de tipo *How*. Esto no supone ningún problema, ya que se amplía la variedad de preguntas generadas sin perder las otras y los "nuevos" tipos de pregunta generados no son sumamente complejos. De hecho, en el caso de *How*, también es un tipo de pregunta usual del inglés y por lo tanto es deseable generar preguntas de este tipo.

El conjunto de posibles etiquetas considerado es entonces ARG0, ARG1, ARG2, ARG3, ARG4, ARGM-TMP, ARGM-LOC. ARG0 es denominado "proto-agente" (responsable de un evento) y suele responder a preguntas de tipo *Who*. ARG1 es denominado "proto-tema" (quien es afectado por el evento) y responde mayoritariamente a preguntas de tipo *What*. En el caso de ARG2 y ARG3, varían su rol dependiendo del predicado, por lo que responden preguntas de distintos tipos (*Who*, *What* y *How*). ARG4 representa el destino de un evento, por lo que responde a preguntas de tipo *Where*. ARGM-TMP y ARGM-LOC representan modificadores de tiempo y de lugar, y responden preguntas de tipo *When* y *Where*, respectivamente.

Texto	Candidato a respuesta	Rol semántico
The mom takes Haley to a doctor	The mom	ARG0
The mom takes Haley to a doctor	Haley	ARG1
He gives her a new toothbrush	her	ARG2
She came from her house	from her house	ARG3
I went to the doctor	to the doctor	ARG4
She does this for three days	for three days	ARGM-TMP
Barack Obama was born in Hawaii	in Hawaii	ARGM-LOC

Tabla 9: Ejemplo de respuestas y su rol semántico

En la sección 5.2 se realiza una evaluación de la selección de respuestas comparando conjuntos de diferentes combinaciones de las etiquetas ya descritas, para, en base a los resultados obtenidos, seleccionar el conjunto que se integrará a la versión final de la herramienta.

4.3. Generación de preguntas

La etapa anterior genera candidatos a respuestas sobre el texto original (con las correferencias resueltas). Esta etapa del pipeline genera una o más preguntas sobre cada candidato a respuesta recibido.

Se evaluaron diversos modelos basados en redes neuronales y se terminó experimentando con dos de ellos: T5 y GPT-2.

Modelo T5

El modelo de T5 en esta etapa recibe una lista donde cada elemento es una copia del texto original luego del preprocesamiento con una etiqueta <hl> alrededor de una de las respuestas elegidas por la etapa anterior. El modelo luego genera una o más preguntas por cada elemento de esta lista. En la tabla 10 se muestran ejemplos del formato del texto que se obtiene como entrada en esta etapa.

Elemento	Candidato a respuesta	Rol semántico
<hl> Adriana <hl> is tall. Adriana has brown hair. Mathew is eleven. Mathew likes basketball.	Adriana	ARG0
Adriana is tall. Adriana has <hl> brown hair <hl>. Mathew is eleven. Mathew likes basketball.	brown hair	ARG1
Adriana is <hl> tall <hl>. Adriana has brown hair. Mathew is eleven. Mathew likes basketball.	tall	ARG2

Tabla 10: Formato en el que se encuentra el texto y las respuestas con los que se entrenó el modelo T5

El modelo base utilizado es un modelo basado en T5 [42] pre entrenado sobre SQuAD para la tarea de generación de preguntas. Este modelo se basa en la variante small de T5 de 60 millones de parámetros y está disponible en *HuggingFace* [50] como *t5-small-qg-hl* [38]. De ahora en adelante tomamos este modelo como línea base y lo denominamos "T5-Base".

A este modelo se le realizó un proceso de fine-tuning con los corpus de SQuAD y NewsQA, sobre el preentranamiento sobre SQuAD que ya tenía el modelo base. Más específicamente, se generan dos modelos distintos según qué proceso de fine-tuning se le aplique:

- Fine-tuning solamente en NewsQA
- Fine-tuning en un corpus resultado de combinar SQuAD y NewsQA

A estos modelos los llamaremos "T5-NewsQA" y "T5-SQuAD y NewsQA", respectivamente.

Para entrenar los modelos se utilizó la biblioteca de PyTorch[37], y código de Suraj Patil[38]. Los hiperparámetros utilizados en el entrenamiento de estos modelos fueron los siguientes:

- *Learning rate*: 10^{-4}
- Tamaño de *batch*: 16
- Cantidad de *epochs*: 3
- *Steps* de acumulación de gradiente: 8
- Semilla (*seed*): 42

Como ya se explicó, SQuAD y NewsQA fueron los corpus utilizados para entrenar los modelos presentados, dado que los otros que se mencionan en la sección 3.2 no poseen el tamaño necesario para ser utilizados para este fin, por lo que se utilizaron solamente con fines de evaluación.

Modelo GPT-2

En el caso de los modelos de GPT-2, se utilizó la variante *small* de 124 millones de parámetros. Se entrenaron dos modelos, uno con SQuAD y otro con NewsQA, a los que denominaremos GPT2-SQuAD y GPT2-NewsQA, respectivamente. Para el entrenamiento de los modelos se utilizó el paquete de Python *aitextgen* [1], que es una herramienta específica para el entrenamiento y generación de texto utilizando GPT-2. Dentro del marco de esta herramienta, los hiperparámetros utilizados fueron:

- Cantidad de *steps*: 25000
- *Learning rate*: 10^{-3}
- Tamaño de *batch*: 1

Previo al entrenamiento, se adaptan los textos de los datasets con los que se va a entrenar para que tengan el siguiente formato:

Contexto <|startofanswer|> Respuesta <|startofquestion|> Pregunta

Donde "<|startofanswer|>" y "<|startofanswer|>" son etiquetas delimitadoras. Luego, con los textos en este formato es que se realiza el entrenamiento, con el fin de que GPT-2 aprenda esta representación, y luego pasándole un texto en esa representación, pero que termina en la etiqueta "<|startofquestion|>", el modelo sea capaz de completar lo que falta con una pregunta.

Una vez entrenados, los modelos de GPT-2 reciben una serie de *prompts*, cada uno de los cuales consiste en el texto original seguido de la etiqueta "<|startofanswer|>", la respuesta deseada (presente en el texto) y la etiqueta "<|startofquestion|>". Es decir que cada *prompt* sigue el siguiente formato:

Contexto <|startofanswer|> Respuesta <|startofquestion|>

Para cada texto se tienen tantas *prompts* como preguntas se quiera generar sobre él y para cada *prompt* se genera una pregunta que tenga como respuesta aquella indicada luego de la etiqueta "<|startofanswer|>". Luego, se obtiene la pregunta extrayendo desde el fin de la etiqueta "<|startofquestion|>" hasta el primer signo de interrogación o fín de línea.

Una ventaja que tiene este formato es que se puede especificar el tipo de pregunta que se desea generar, agregándole la palabra de pregunta (*question word*) inmediatamente luego de la etiqueta "<|startofquestion|>".

4.4. Postprocesamiento

En la etapa de postprocesamiento se resuelven otros inconvenientes presentados por la salida de las etapas anteriores.

Se observó que muchas de las preguntas generadas por este estaban malformadas; algunas contenían múltiples signos de interrogación mientras que otras en realidad eran varias preguntas en sucesión separadas por puntos o signos de interrogación.

Para lidiar con este problema, se decidió que las preguntas generadas por la etapa anterior fueran procesadas nuevamente:

- Para las preguntas con múltiples signos de interrogación, se eliminaron todos salvo el primero.
- Para las preguntas que eran varias preguntas separadas por signos de interrogación, se eliminó todo lo posterior al primer signo.

5. Evaluación experimental

Se realizaron distintos tipos de evaluaciones. Por un lado se realizó una evaluación midiendo las métricas BLEU, METEOR o ROUGE para la generación de preguntas y la otra consistió en medir valores de Precision, Recall, Accuracy y F1 para selección de respuestas. Además, para la ejecución del *pipeline* completo, se realiza una evaluación *exact match* y se calculan las mismas métricas que para la selección de respuestas. Para calcular la métricas de BLEU, METEOR y ROUGE se utilizó el código de Sharma et al.[45]. Para calcular las métricas de Precision, Recall, Accuracy y F1 se utilizó la biblioteca de *scikit-learn* [39].

5.1. Generación de preguntas

Para la parte de generación de preguntas se evaluaron modelos basados en T5 y otros en GPT-2. En ambos casos estos modelos pasan por un proceso de fine-tuning sobre los corpus de SQuAD y/o NewsQA, como se mencionó en la sección 4.3.

Por lo tanto, los distintos modelos evaluados son:

- T5-Base (pre entrenado en SQuAD sin finetuning)
- T5-NewsQA (T5-Base con fine-tuning en NewsQA)
- T5-SQuAD y NewsQA (T5-Base con fine-tuning en un corpus que une SQuAD y NewsQA)
- GPT2-NewsQA (GPT2 base entrenado con NewsQA)
- GPT2-SQuAD (GPT2 base entrenado con SQuAD)

Para la realización de pruebas se dispone de los conjuntos de datos presentados en la sección 3. Utilizamos el corpus 2019+2021 para realizar una evaluación en un conjunto que representa el nivel de inglés que se busca, y utilizamos los corpus de NewsQA y SQuAD para obtener resultados en corpus de uso general y tener un punto de comparación con trabajos previos.

En el caso de NewsQA, se evalúa sobre el conjunto de test incluido en el propio dataset. En el caso de SQuAD, se utilizó su versión 1.1 con una división train-dev-test presentada en [17], que es utilizada por distintas publicaciones. Esto es debido a que el conjunto original de SQuAD no provee un conjunto de test.

Para la evaluación, se les provee como entrada a los distintos modelos tanto los textos que componen el conjunto de evaluación de los distintos corpus, como las respuestas para cada uno de estos textos. Después se observan las preguntas que cada modelo genera como salida. Estas preguntas luego se comparan con las preguntas contenidas en el conjunto de evaluación y se toman las métricas que se quieren evaluar.

Se evalúa el conjunto de preguntas generadas en base a las métricas BLEU, METEOR y ROUGE. En el caso de BLEU se toman las métricas de BLEU-1, BLEU-2, BLEU-3 y BLEU-4 mientras que de ROUGE se toma la métrica ROUGE-L. Cabe destacar que

BLEU-4 es a veces denominada BLEU por NLTK y se calcula en base a los n-gramas de 1 a 4, utilizando pesos uniformes [2].

Modelo	$BLEU_1$	$BLEU_2$	$BLEU_3$	$BLEU_4$	$METEOR$	$ROUGE_L$
T5-Base	0.496	0.424	0.365	0.308	0.427	0.683
T5-NewsQA	0.415	0.359	0.307	0.256	0.409	0.629
T5-SQuAD y NewsQA	0.323	0.275	0.231	0.188	0.378	0.536
T5-Base + Coref	0.569	0.486	0.415	0.351	0.430	0.679
T5-NewsQA + Coref	**0.724**	**0.646**	**0.575**	**0.508**	**0.468**	**0.772**
T5-SQuAD y NewsQA + Coref	0.563	0.488	0.420	0.355	0.441	0.699
GPT2-NewsQA	0.054	0.019	0.006	5.475e-07	0.073	0.084
GPT2-SQuAD	0.0310	0.0145	0.0073	0.003	0.062	0.148

Tabla 11: Métricas de los distintos modelos sobre el Dataset 2019+2021

En la tabla 11 se muestra la comparación de las métricas obtenidas por los diferentes modelos sobre el dataset 2019+2021, en particular se puede ver que los modelos basados en T5 en general tuvieron un buen desempeño mientras que aquellos basados en GPT-2 obtuvieron muy malos resultados.

Además, se puede observar la comparación entre realizar o no el paso de resolución de correferencias sobre el texto antes de generar las preguntas. Podemos notar lo importante que resulta este proceso, dado que aumentan los valores de todas las métricas, y muchas de ellas de manera considerable.

El proceso de resolución de correferencias también genera que el modelo de T5 con mejores resultados cambie: previo a este proceso el modelo T5-Base es el que obtiene un mejor rendimiento, pero al resolver las correferencias, el modelo al que se le realizó finetuning con NewsQA (T5-NewsQA) pasa a ser el que obtiene mejores resultados sobre este dataset.

Es de notar el hecho de que al obtener buenos resultados sobre nuestro dataset, que contiene un nivel de inglés adecuado a lo que se busca (A1 y A2), quiere decir que los modelos que entrenamos son capaces de generar preguntas de nivel A1 y A2, si son provistas por textos con el mismo nivel.

Con esto en cuenta, el modelo que más se ajusta a las necesidades presentes es el modelo T5-NewsQA, dado que es el que mejor rendimiento ofrece sobre el dataset del que más importan los resultados, ya que es el que contiene preguntas del tipo y dificultad buscadas.

Modelo	$BLEU_1$	$BLEU_2$	$BLEU_3$	$BLEU_4$	$METEOR$	$ROUGE_L$
T5-Base	0.179	0.107	0.0694	0.047	0.179	0.235
T5-NewsQA	0.282	0.186	0.129	0.093	0.196	0.307
T5-SQuAD y NewsQA	0.264	0.172	0.119	0.0844	0.179	0.290

Tabla 12: Métricas de los distintos modelos sobre NewsQA

Modelo	$BLEU_1$	$BLEU_2$	$BLEU_3$	$BLEU_4$	$METEOR$	$ROUGE_L$
T5-Base	0.356	0.252	0.189	0.146	0.231	0.360
T5-NewsQA	0.335	0.235	0.175	0.135	0.208	0.350
T5-SQuAD y NewsQA	0.369	0.265	0.201	0.156	0.246	0.384

Tabla 13: Métricas de los distintos modelos sobre SQuAD

En las tablas 12 y 13 se muestran los resultados obtenidos por los modelos entrenados sobre

T5 para los conjuntos de datos de evaluación de NewsQA y SQuAD, respectivamente. Se descartó realizar una evaluación sobre los modelos entrenados basados en GPT-2 dado su mal desempeño sobre nuestro dataset.

Se puede observar que el modelo que mejor desempeño presenta para las evaluaciones sobre los distintos datasets es aquel al que se le realizó *finetuning* con el conjunto de entrenamiento del mismo dataset. Es decir, el modelo con mejores resultados para NewsQA fue aquel con *finetuning* en NewsQA y el modelo con mejores resultados para SQuAD fue aquel con *finetuning* en NewsQA y SQuAD. Esto se puede explicar con el hecho de que al entrenar un modelo con un determinado corpus, se adapta al tipo de preguntas que contiene, por lo que al evaluar en el mismo corpus, tiene sentido que obtenga mejores resultados que otro modelo entrenado con un corpus distinto.

También se puede observar como los resultados son bastante menores que los obtenidos sobre nuestro dataset. Esto principalmente se debe a que NewsQA y SQuAD contienen textos de mayor complejidad por lo que las preguntas generadas también son más complejas y esto induce a una mayor cantidad de errores al generarlas.

Modelo	$BLEU_4$	$METEOR$	$ROUGE_L$
T5-NewsQA	13.45	20.80	35.02
T5-SQuAD y NewsQA	16.32	24.55	38.38
Du et al.	12.28	16.62	39.75
Du and Cardie	15.16	19.12	No reportado
Song et al.	13.98	18.77	42.72
Liu et al.	17.55	21.24	44.53
Dong et al.	22.12	25.06	51.07

Tabla 14: Métricas de los distintos modelos sobre SQuAD

En la tabla 14 se muestran los resultados de los dos modelos entrenados sobre T5 comparados con los resultados reportados por los trabajos mencionados en la sección 2.2.5. Se incluyeron sólo resultados de trabajos que evalúan sobre el mismo split de SQuAD [17], por lo que los resultados reportados por Zhou et al. [52] no se incluyen. Cabe destacar que los resultados presentes en la tabla son los mejores resultados reportados por cada trabajo.

Se puede observar que se logra un desempeño igual o incluso superior que algunas de las otras herramientas, principalmente a la de los trabajos anteriores a 2019, mientras que en otros casos, principalmente Dong et al., se está lejos de lograr un desempeño similar. Esto en principio no supone un mayor inconveniente, ya que el objetivo nunca fue lograr superar los resultados obtenidos para estos antecedentes sobre SQuAD, especialmente considerando que este dataset presenta una complejidad superior a la que está destinada la herramienta.

Es importante mencionar que al evaluar sobre SQuAD, no realizamos el proceso de resolución de correferencias sobre los textos del dataset. Esto es porque los textos de SQuAD presentan una mayor complejidad y es probable que resolver las correferencias introduzca más errores que en textos simples como los de nuestro dataset. Sin embargo esto podría llegar a mejorar los resultados obtenidos.

Concluyendo, se tiene que el modelo "T5-SQuAD y NewsQA" obtiene resultados bastante competitivos en SQuAD con los reportados por otros trabajos relacionados, y dentro de nuestros modelos, es el que presenta mejor desempeño al evaluar sobre SQuAD. Sin embargo, el modelo que se ajusta más a las necesidades del proyecto es el "T5-NewsQA", dado que obtiene resultados ligeramente peores en SQuAD, pero es el modelo que mejor desempeño presenta en nuestro dataset, que es algo que consideramos de mayor importancia.

5.2. Selección de respuestas

Para elegir los candidatos a respuesta se corre el selector de respuestas planteado en la sección 4.2, utilizando distintos conjuntos de etiquetas, sobre los textos del conjunto de evaluación y se evalúan las métricas de *Precision*, *Recall*, *Accuracy* y *F1* frente a las respuestas de referencia.

El conjunto de evaluación utilizado para esta prueba es el conjunto combinado entre el corpus de 2019 y el de 2021. Se evalúa sobre este conjunto puesto que al tener un tamaño no muy grande se pueden observar manualmente las salidas del selector de respuestas, y analizar los errores obtenidos. Además, este corpus está más alineado con el objetivo de este trabajo mientras que conjuntos como SQuAD y NewsQA son de uso más general y podrían tener respuestas que generen preguntas de un nivel de Inglés más avanzado que el buscado.

En este caso no se tienen distintos selectores de respuestas para comparar entre sí, pero sí se evaluaron distintos conjuntos de roles semánticos a considerar como posibles candidatos a respuestas. Estos distintos conjuntos son:

- Set1: {ARG0, ARG1}
- Set2: {ARG0, ARG1, ARG2}
- Set3: {ARG0, ARG1, ARG2, ARG3, ARG4}
- Set4: {ARG0, ARG1, ARG2, ARGM-TMP, ARGM-LOC}
- Set5: {ARG0, ARG1, ARG2, ARG3, ARG4, ARGM-TMP, ARGM-LOC}

	Precision	Recall
Set 1	0.58	0.71
Set 2	0.52	0.77
Set 3	0.52	0.78
Set 4	0.51	0.79
Set 5	0.51	0.81

Tabla 15: Métricas obtenidas por el selector de respuestas para los distintos conjuntos de roles semánticos sobre el Dataset 2019+2021

Las métricas obtenidas para cada uno de los conjuntos de roles semánticos se pueden ver en la tabla 15. Como se puede observar, el set 1 obtuvo los mejores resultados en todas las métricas salvo en *Recall*, medida en la que se mejora cada vez que se agranda el conjunto. Esto tiene sentido pues al agregar elementos al conjunto se generan más variedad

de respuestas y esto hace que se logre cubrir más cantidad del conjunto de referencia, sin perder ninguna respuesta generada antes de agrandar el conjunto. A su vez es lógico que baje la *Precision* por la misma razón, al generar más tipos de pregunta es más probable que suba la cantidad de elementos que no estén en el conjunto de referencia.

Por otro lado, la Accuracy y la medida de F1, salvo una muy leve diferencia en el Set 1, se mantiene pareja para todos los conjuntos de roles semánticos considerados.

Para la elección del conjunto de etiquetas que se utilizó en la herramienta final, nos basamos principalmente en la métrica de *Recall*. Esto es porque lo que más nos interesa es generar una buena variedad de preguntas y cubrir lo más posible el conjunto de preguntas de referencia. Además, como el valor de la *Precision* varía poco entre los distintos conjuntos, significa que no se están generando muchas preguntas de sobra al aumentar el tamaño del conjunto de etiquetas.

Por estas razones, se elige el conjunto de etiquetas {ARG0, ARG1, ARG2, ARG3, ARG4, ARGM-TMP, ARGM-LOC} para implementar en la herramienta final.

5.3. Evaluación completa

Hasta ahora en las secciones 5.1 y 5.2 presentamos los resultados obtenidos para las etapas de selección de respuestas y de generación de preguntas de forma aislada. En esta sección se presentan resultados de la evaluación de las preguntas generadas por la herramienta al ejecutar el pipeline completo. Esto es, dado un texto de entrada, aplicarle la resolución de correferencias, luego seleccionar los candidatos a respuestas, generar las preguntas en base a las respuestas obtenidas y realizar la etapa de postprocesamiento a las preguntas generadas.

Para esta evaluación, se utilizó la técnica de *exact match*, que consiste en, dados dos conjuntos de preguntas, comparar la cantidad de preguntas que coinciden exactamente y las que no entre el conjunto de preguntas generadas por la ejecución completa del pipeline y el de referencia. Luego, a partir de esto, se calculan las métricas de *Precision*, *Recall*, *Accuracy* y *F1*.

	Precision	Recall	Accuracy	F1
Pipeline1	0.31	0.47	0.23	0.37
Pipeline2	0.25	0.39	0.18	0.31

Tabla 16: Métricas obtenidas obtenidas por el *pipeline* completo sobre el dataset 2019+2021

En la tabla 16 se muestra los resultados obtenidos de las métricas para el *pipeline* entero, utilizando el conjunto de etiquetas {ARG0, ARG1, ARG2, ARG3, ARG4, ARGM-TMP, ARGM-LOC} en la etapa de selección de respuestas. Los resultados de *Pipeline1* son obtenidos utilizando el modelo T5-NewsQA en la etapa de generación de preguntas, mientras que para *Pipeline2* se utilizó el modelo T5-NewsQA y SQuAD para la misma etapa. Podemos observar que al igual que en la sección 5.1, el modelo que mejores resultados obtiene dentro de nuestro corpus es el T5-NewsQA.

Para el *Pipeline1*, podemos observar que alrededor de un 31 % de las preguntas generadas se encuentran también en el conjunto de referencia (*Precision*) y aproximadamente un 47 % de las preguntas del conjunto de referencia son cubiertas por las preguntas generadas *Recall*. Esto quiere decir que aproximadamente un 69 % de las preguntas generadas están "de sobra" pero se logra cubrir de forma exacta casi la mitad del conjunto de referencia. Esto tiene sentido dada la prioridad que se le dio a la métrica de Recall a la hora de elegir el conjunto de etiquetas usado en la etapa de selección de respuestas.

Cabe mencionar que estos números parten de un *exact match*, por lo que pueden haber muchas preguntas que sean casi iguales, representen lo mismo y tengan la misma respuesta, pero no estén consideradas dentro de estos resultados. Un ejemplo de esto es que el *pipeline* generó la pregunta "*When was Obama elected president*?" mientras que en las preguntas de referencia la pregunta que está es "*When was Barack Obama elected president*?". Además hay preguntas generadas que no están en las de referencia pero que en verdad son perfectamente factibles. Por ejemplo, para la oración "*My favorite beach is called Emerson Beach.*" perteneciente a uno de los textos de referencia, se genera la pregunta "*What is my favorite beach called?*", que no está en el conjunto de preguntas de referencia pero es una pregunta totalmente válida.

Por otro lado, también se debe notar que se está evaluando sobre el *pipeline* entero, por lo que los errores de cada etapa se propagan a las siguientes. Por ejemplo, si una respuesta candidata no es generada por la etapa de selección de respuestas, esto hace que la pregunta generada por esa respuesta, que está en el conjunto de referencia, no sea generada por la etapa de generación de preguntas.

Tipo	En corpus	Generadas	Correctas	Precision	Recall
Who	175	239	105	0.44	0.60
Where	38	33	7	0.21	0.18
What	218	400	111	0.28	0.51
When	9	33	1	0.03	0.11
How	14	14	0	0.00	0.00

Tabla 17: Resultados obtenidos para cada tipo de pregunta sobre dataset 2019+2021

En la tabla 17 se muestran los resultados obtenidos por el *pipeline1* para distintos tipos de pregunta dentro de nuestro corpus. Se puede observar como las preguntas que mejor genera el modelo son las de tipo *Who*, seguidas de las de tipo *What*, que además son las dos de mayor presencia dentro del dataset. Por otro lado, se obtienen resultados bastante malos para las preguntas de tipo *When* y *How*, que son las que menos aparecen en el corpus. De esto se podría establecer una relación entre el porcentaje del corpus que abarca un tipo de pregunta, y los resultados que se obtienen para ese tipo.

En cuanto a las preguntas generadas, se puede notar que más de la mitad son de tipo *What*, aproximadamente un tercio de las preguntas son de tipo *Who* y el resto está comprendido por las de tipo *Where*, *When* y *How*. Un elemento que influye en estas cantidades es el dataset con el que fue entrenado el modelo. Como mostramos en la sección 3.1, los corpus de SQuAD y NewsQA contienen en su mayoría preguntas de tipo *What*. En el caso de NewsQA, las preguntas de tipo *Who* y *When* abarcan el mismo porcentaje, que a su

vez es bastante mayor que el de las de tipo *Where* y *How*. Por el lado de SQuAD, estos cuatro tipos de pregunta abarcan porcentajes similares. Con esto dicho, tiene sentido que las cantidades sean las de la tabla, dado que es esperable que las preguntas generadas presenten una distribución similar a los porcentajes de cada tipo de pregunta presentes en el dataset con el que se entrenó el modelo,

Otro factor que influye en la cantidad de preguntas generadas de cada tipo, y posiblemente la razón por la que se generan mucho más preguntas de tipo *Who* que los demás tipos, salvo *What*, son las características de los textos comprendidos en el dataset sobre el que evaluamos. Nuestro dataset (2019+2021) está comprendido por textos y preguntas de un nivel básico de inglés, en contraparte con los datasets de NewsQA y SQuAD, que son más complejos. Por esto, los textos de nuestro dataset, al ser más simples, pueden prestarse a que se genere una mayor cantidad de preguntas de tipo *Who*.

Además, es importante decir que el hecho de que una pregunta generada no esté en el corpus, no significa que esta pregunta sea incorrecta, solamente quiere decir que no fue anotada al desarrollar el corpus. También puede ser que sea una pregunta que tenga una ligera variación con respecto a alguna de las pertenecientes al corpus, pero al no ser idénticas, la pregunta generada se cuenta como incorrecta.

Para concluir con esta sección, presentamos una comparativa de nuestro trabajo con el proyecto de grado de 2019, con el fin de poder observar la diferencia que se obtiene al utilizar distintos enfoques. Cabe destacar que en el proyecto de 2019, se evaluó sobre un fragmento equivalente al 20 % del corpus que se utilizó para ese trabajo. Al no tener esta división, nosotros evaluamos sobre el dataset entero.

Tipo	En corpus	Generadas	Correctas	Precision	Recall
Who	47	48	40	0.833	0.851
Where	4	2	1	0.500	0.251
What	33	34	28	0.824	0.849
When	1	2	1	0.500	1.000
What colour	1	1	1	1.000	1.000

Tabla 18: Resultados reportados en el proyecto de grado de 2019

En la tabla 18 se pueden ver los resultados reportados en el proyecto de 2019, sobre la división mencionada. De esta tabla podemos calcular los valores de *Precision* y *Recall* sobre todas las preguntas (sin dividir por tipo). Estos son de 0.816 y 0.826 respectivamente.

En la tabla 19 se muestran los valores de *Precision* y *Recall* obtenidos por el *Pipeline1* de la tabla 16, sobre el corpus construido en el proyecto de 2019. Podemos ver que son bastante menores que los obtenidos en el proyecto de 2019. Esto principalmente se debe a que en un enfoque basado en reglas las preguntas generadas todas tienen una estructura similar, por lo que es más sencillo lograr replicar de forma casi exacta el conjunto de

	Precision	Recall
Pipeline1	0.42	0.58

Tabla 19: Resultados obtenidos por nuestra herramienta en el corpus construido en el proyecto de grado de 2019

referencia.

Además, como mencionamos anteriormente, el conjunto de evaluación no es el mismo en ambos casos, sino que nuestro modelo es evaluado en un conjunto más grande que incluye al que se utilizó para la evaluación en el proyecto de 2019. De igual manera, esto no debería implicar una diferencia mayor en los resultados, pero es algo a tener en cuenta.

También es importante mencionar que el valor de *Recall* obtenido se ve disminuido por las preguntas que el modelo genere que no sean de uno de los cinco tipos listados en la tabla 18, dado que el conjunto de referencia solo tiene preguntas de estos tipos. Esto se ve acentuado por la decisión mencionada en la sección 5.2, de tomar el conjunto de etiquetas que produzca una variedad más amplia de preguntas. En contraparte, en el proyecto de 2019, al utilizar un enfoque basado en reglas, solamente se generan preguntas de los tipos que se encuentran en el conjunto de referencia.

6. Conclusiones y trabajo futuro

En base a los objetivos planteados inicialmente, se logró el objetivo principal del desarrollo de una herramienta que permite generar preguntas de manera automática a partir de un texto de inglés. Además, se cumplió con el objetivo de que dado un texto con el nivel de inglés adecuado (A1 o A2), las preguntas generadas sean del mismo nivel.

Se logró entrenar y evaluar distintos modelos para la generación de preguntas, pudiendo usar dos arquitecturas distintas (T5 y GPT-2), obteniendo, en casi todos los casos, resultados satisfactorios. No se llegó a un buen resultado con respecto a los modelos entrenados sobre GPT-2, pero no se descarta que esto fuera por un tema de limitación de capacidad de cómputo. Además, se pudo realizar una comparación con trabajos anteriores de generación de preguntas utilizando modelos neuronales, lo que mostró que los modelos entrenados son bastante competitivos en cuanto a los resultados que se obtuvieron.

Se consiguió desarrollar una herramienta para la extracción de respuestas de un texto, basada en *semantic role labeling*, que obtuvo buenos resultados al evaluar sobre el dataset 2019+2021.

También se consiguió expandir el corpus de preguntas y respuestas construido por los autores del proyecto de 2019 con nuevos textos y sus respectivos pares de preguntas y respuestas. Este corpus fue fundamental a la hora de evaluar los modelos generados y con ello corroborar que las preguntas generadas presentan la calidad esperada.

Se pudo continuar con la línea de trabajo que marcó el proyecto 2019, explorando un método distinto como lo es la utilización de redes neuronales, obteniendo buenos resultados y ofreciendo una alternativa más escalable y fácilmente comparable con otros trabajos.

Por último, se cumplió con el objetivo de profundizar en el área del Procesamiento del Lenguaje Natural, logrando nuevos conocimientos, estudiando trabajos previos, explorando técnicas y herramientas de común uso en el área y adquiriendo experiencia en el manejo de ellas.

En cuanto al trabajo futuro, se presentan distintas líneas de trabajo para mejorar el rendimiento de la herramienta, o bien su disponibilidad.

- Experimentar con otros modelos de lenguaje como BERT o XLNet.
- Seguir expandiendo el corpus de preguntas y respuestas presentado en el proyecto de 2019.
- Integrar la herramienta desarrollada en una plataforma web, que puede ser en la generada en el proyecto de 2019.
- En caso de conseguir un corpus de suficiente tamaño, con un nivel de inglés A1 y A2, ya sea externo o expandiendo lo suficiente el presentado en este trabajo, utilizarlo para el entrenamiento o finetuning de los modelos.
- Relevar el uso de distintas técnicas y herramientas para la selección de respuestas. Se podría experimentar con una selección de respuesta neuronal o con un enfoque

neuronal que genere los pares de preguntas y respuestas simultáneamente.

- En el caso de conseguir plataformas con buena capacidad computacional, realizar un entrenamiento más extenso de los modelos o utilizando variantes más grandes de los modelos.
- Implementar un sistema de ranking de preguntas similar al planteado en el proyecto de 2019.
- Implementar un sistema de evaluación de las preguntas generadas, que indique si la pregunta es acorde o no a un nivel de inglés A1/A2.
- Experimentar con otras formas de entrenamiento con GPT-2, puede ser cambiando el modo en que se entrena utilizando oraciones como contexto en lugar del texto entero.

Referencias

[1] Aitextgen. https://docs.aitextgen.io/. Último acceso: 07/04/2022.

[2] Bleu score, nltk. https://www.nltk.org/_modules/nltk/translate/bleu_score.html. Último acceso: 10/04/2022.

[3] Inglés sin límites, anep. https://www.anep.edu.uy/codicen/politicas-linguisticas/focus-on-first. Último acceso: 10/04/2022.

[4] Lingua.com reading texts. https://lingua.com/english/reading/. Último acceso: 03/04/2022.

[5] Oludare Abiodun, Aman Jantan, Oludare Omolara, Kemi Dada, Nachaat Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4:e00938, 11 2018.

[6] Saahil Afaq and Smitha Rao. Significance of epochs on training a neural network. *International Journal of Scientific & Technology Research*, 9:485–488, 2020.

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2014.

[8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[9] Claire Bonial, Olga Babko-Malaya, Jinho D. Choi, and Jena D. Hwang. Propbank annotation guidelines. 2010.

[10] Stevo Bozinovski. Reminder of the first paper on transfer learning in neural networks, 1976. *Informatica*, 44, 09 2020.

[11] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014.

[12] Augusto Cortez Vásquez, Hugo Vega huerta, Jaime Pariona Quispe, and Ana Maria Huayna. Procesamiento de lenguaje natural. *Revista de investigación de Sistemas e Informática*, 6(2):45–54, dic. 2009.

[13] Rubel Das, Antariksha Ray, Souvik Mondal, and Dipankar Das. A rule based question generation framework to deal with simple and complex sentences. In *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 542–548, 2016.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[15] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation, 2019.

[16] Xinya Du and Claire Cardie. Harvesting paragraph-level question-answer pairs from Wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1907–1917, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[17] Xinya Du, Junru Shao, and Claire Cardie. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[18] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. Allennlp: A deep semantic natural language processing platform. 2017.

[19] Daniel Gildea and Martha Palmer. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 239–246, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[20] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany, August 2016. Association for Computational Linguistics.

[21] Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. Pointing the unknown words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 140–149, Berlin, Germany, August 2016. Association for Computational Linguistics.

[22] Michael Heilman. *Automatic factual question generation from text.* PhD thesis, Carnegie Mellon University, 2011.

[23] Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition.* Pearson Prentice Hall, Upper Saddle River, N.J., 2009.

[24] Dan Jurafsky and James H. Martin. Speech and language processing. Draft de tercera edición disponible en https://web.stanford.edu/~jurafsky/slp3/, December 2021.

[25] Nikhil Ketkar. *Stochastic Gradient Descent*, pages 113–132. Apress, Berkeley, CA, 2017.

[26] Paul Kingsbury and Martha Palmer. From TreeBank to PropBank. In *Proceedings of the Third International Conference on Language Resources and Evaluation*

(LREC'02), Las Palmas, Canary Islands - Spain, May 2002. European Language Resources Association (ELRA).

[27] Alon Lavie and Michael J. Denkowski. The meteor metric for automatic evaluation of machine translation. *Machine Translation*, 23(2–3):105–115, sep 2009.

[28] Kenton Lee, Luheng He, and L. Zettlemoyer. Higher-order coreference resolution with coarse-to-fine inference. In *NAACL-HLT*, 2018.

[29] Chin-Yew Lin and Franz Josef Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, page 605–es, USA, 2004. Association for Computational Linguistics.

[30] Bang Liu, Mingjun Zhao, Di Niu, Kunfeng Lai, Yancheng He, Haojie Wei, and Yu Xu. Learning to generate questions by LearningWhat not to generate. In *The World Wide Web Conference on - WWW '19*. ACM Press, 2019.

[31] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[32] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[33] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2 edition, 2018.

[34] Martín Jaime Morón and Joaquín Scocozza Díaz. *Construcción de herramientas para enseñanza de inglés: generación de preguntas y respuestas*. Tesis de grado, Universidad de la República (Uruguay). Facultad de Ingeniería, 2020.

[35] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Comput. Linguist.*, 31(1):71–106, mar 2005.

[36] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics.

[37] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[38] Suraj Patil. Question generation. https://github.com/patil-suraj/question_generation, 2020.

[39] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[40] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[41] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[42] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[43] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

[44] Yutaka Sasaki. The truth of the f-measure. *Teach Tutor Mater*, 01 2007.

[45] Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799, 2017.

[46] Peng Shi and Jimmy Lin. Simple bert models for relation extraction and semantic role labeling. *ArXiv*, abs/1904.05255, 2019.

[47] Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[48] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[50] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[51] Shiyue Zhang and Mohit Bansal. Addressing semantic drift in question generation for semi-supervised question answering, 2019.

[52] Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and M. Zhou. Neural question generation from text: A preliminary study. In *NLPCC*, 2017.