

Generación de Preguntas a partir de Transcripciones utilizando Facebook/opt-350

Eduards Mendez , Daniel Meyer
Universidad Nacional de Colombia
emendezc@unal.edu.co, gdmeyerp@unal.edu.co

Abstract—En este artículo se presenta un ejemplo de cómo generar preguntas a partir de una transcripción utilizando el modelo Facebook/opt-350. Este enfoque está diseñado para ser compacto y fácil de depurar, permitiendo la segmentación de textos y la creación de preguntas automáticas basadas en el contenido transcrito.

Index Terms—Segmentación de textos, NLTK, Facebook/opt-350, embeddings, transcripción, generación de preguntas.

I. INTRODUCCIÓN

Este documento presenta un método para generar preguntas a partir de transcripciones de audio utilizando el modelo Facebook/opt-350. El objetivo es proporcionar una herramienta eficiente y fácil de usar que permita segmentar textos y generar preguntas automáticas basadas en el contenido de las transcripciones. Este enfoque es especialmente útil para aplicaciones que requieren el análisis y procesamiento de grandes cantidades de texto transcrito, como en el caso de videos educativos, entrevistas, o conferencias.

II. CARACTERÍSTICAS PRINCIPALES

- **Segmentación de textos con NLTK:** Utiliza NLTK para dividir el texto en segmentos manejables. Esto facilita la identificación de los puntos clave en el texto y mejora la precisión de las respuestas generadas.
- **Modelo Facebook/opt-350m:** Implementa el modelo para la generación de preguntas y respuestas. Este modelo es capaz de manejar grandes volúmenes de datos y proporciona respuestas coherentes y contextualmente relevantes.
- **Embeddings y búsqueda de similitud:** Emplea embeddings de HuggingFace y el algoritmo Annoy para encontrar los segmentos más relevantes del texto. Esto asegura que las preguntas sean respondidas con la información más pertinente disponible.

A. Segmentación de textos con NLTK

NLTK (Natural Language Toolkit) es una biblioteca en Python que proporciona herramientas para el procesamiento del lenguaje natural. Utilizamos NLTK para segmentar el texto transcrito en oraciones y párrafos. La segmentación de textos se realiza utilizando los tokenizadores de oraciones y párrafos de NLTK, que dividen el texto en fragmentos más pequeños y manejables, facilitando así la generación de preguntas.

B. Modelo Facebook/opt-350m

OPT fue preentrenado predominantemente con texto en inglés, pero una pequeña cantidad de datos que no están en inglés todavía están presentes dentro del corpus de entrenamiento a través de CommonCrawl. El modelo se preentrenó utilizando un objetivo de modelado de lenguaje causal (CLM). OPT pertenece a la misma familia de modelos solo decodificadores como GPT-3. Como tal, fue entrenado previamente utilizando el objetivo de modelado de lenguaje causal autosupervisado.

Para la evaluación, OPT sigue GPT-3 utilizando sus indicaciones y la configuración experimental general. Para más detalles, por favor lea el papel oficial.

C. Embeddings y búsqueda de similitud

Los embeddings son representaciones vectoriales de palabras o textos que capturan su significado semántico. Utilizamos el modelo "all-MiniLM-L6-v2" de HuggingFace para generar embeddings de alta calidad. Estos embeddings se utilizan en combinación con el algoritmo Annoy (Approximate Nearest Neighbors Oh Yeah) para buscar los segmentos de texto más similares a la pregunta formulada. Annoy es eficiente para buscar vecindarios cercanos en grandes conjuntos de datos, lo que asegura que las respuestas generadas sean relevantes y precisas.

III. INSTALACIÓN DE PAQUETES NECESARIOS

Para utilizar este método, es necesario instalar una serie de paquetes, incluyendo transformers, bitsandbytes, accelerate. Además, se debe instalar Whisper para la transcripción de audio. Las instrucciones detalladas de instalación se presentan a continuación.

A. Instalación de paquetes

Estos paquetes incluyen:

- **Transformers:** Utilizado para implementar y usar modelos de lenguaje preentrenados.
- **Bitsandbytes:** Optimiza el uso de memoria y el rendimiento de los modelos de lenguaje.
- **Accelerate:** Proporciona herramientas para la aceleración de modelos en hardware especializado.

IV. CARGA DEL MODELO Y LIBRERÍAS

Se utilizan los modelos de transformers y el modelo Facebook/opt-350, cargados utilizando PyTorch para permitir

la generación de preguntas y respuestas a partir del texto segmentado.

```
import transformers
from transformers import LlamaTokenizer
from peft import PeftModel
import torch
import accelerate

BASE_MODEL = 'facebook/opt-350m'

tokenizer = AutoTokenizer.from_pretrained(
    BASE_MODEL, use_auth_token=token)

model = AutoModelForCausalLM.from_pretrained(
    BASE_MODEL, torch_dtype=torch.float16,
    device_map="auto", use_auth_token=token)

pipeline = transformers.pipeline(
    "text-generation",
    model=model,
    max_new_tokens=200,
    tokenizer=tokenizer,
    return_full_text=False,
    torch_dtype=torch.float16,
    device_map="auto"
)
```

V. SEGMENTACIÓN DE TEXTOS

La segmentación se realiza utilizando NLTK, dividiendo el texto en oraciones y párrafos para facilitar el procesamiento subsecuente. Esta técnica mejora la eficiencia del modelo al permitirle enfocarse en fragmentos específicos del texto en lugar de analizar el contenido en su totalidad.

A. Asignación del tiempo del texto segmentado

Una vez segmentado el texto, se asignan timestamps a cada sección segmentada para mantener la sincronización con el audio original. Esto es crucial para aplicaciones donde el contexto temporal es importante, como en la generación de subtítulos o en el análisis de discursos.

VI. MODELO DE EMBEDDINGS Y BÚSQUEDA DE SIMILITUD

El modelo de embeddings HuggingFace se utiliza para convertir los textos segmentados en vectores. Estos vectores se utilizan en combinación con el algoritmo Annoy para encontrar los segmentos de texto más similares a la pregunta formulada. Esta búsqueda de similitud asegura que las respuestas generadas sean relevantes y precisas.

A. Embeddings

Los embeddings son representaciones vectoriales de palabras o textos que capturan su significado semántico. Usamos el modelo "all-MiniLM-L6-v2" de HuggingFace para generar embeddings de alta calidad. Estos embeddings permiten que

el modelo capture el contexto semántico de las palabras y frases, lo que es crucial para generar respuestas precisas y contextualmente relevantes.

B. Algoritmo Annoy

Annoy (Approximate Nearest Neighbors Oh Yeah) es un algoritmo eficiente para buscar las vecindades más cercanas en grandes conjuntos de datos. Este algoritmo es ideal para recuperar rápidamente los segmentos de texto más relevantes en función de la similitud de embeddings. Annoy construye árboles de búsqueda que permiten encontrar aproximaciones rápidas a los vecinos más cercanos en espacios de alta dimensión, lo que es esencial para el procesamiento eficiente de grandes volúmenes de datos textuales.

VII. METODOLOGÍA

En esta sección, se describirá la metodología empleada para abordar los dos desafíos mencionados anteriormente: la generación de títulos de capítulos y la respuesta a preguntas basadas en videos de conferencias.

Sin embargo, antes de abordar estos problemas, necesitábamos transcribir los archivos de audio de la conferencia. Para esta tarea, utilizamos el modelo Whisper desarrollado por OpenAI, específicamente optando por la variante "small". Este modelo consumió aproximadamente 2 GB de VRAM y tomó aproximadamente 5 minutos para transcribir toda la conferencia de una hora y media utilizando una sola GPU Nvidia T4. Seleccionamos esta versión del modelo debido a su equilibrio encomiable entre precisión y velocidad de procesamiento. Este modelo no solo produce el texto de la transcripción de la conferencia, sino también las marcas de tiempo de cada segmento individual de la transcripción, que suelen ser de unos 15 a 30 segundos cada uno.

A. Capítulos

En relación con la generación de títulos de capítulos, después de obtener la transcripción, nuestro primer paso fue dividir el texto en segmentos que fueran semánticamente coherentes. Esta segmentación fue necesaria para generar títulos de capítulos significativos a partir de estos segmentos. Para lograr esto, empleamos el algoritmo TextTilingTokenizer proporcionado por el Natural Language Toolkit (nltk). Este algoritmo identifica cambios en los subtemas analizando patrones de co-ocurrencia léxica dentro del texto.

Después del proceso de segmentación, procedimos a emplear el modelo de lenguaje (LLM) para generar títulos de capítulos adecuados que representaran con precisión el contenido de cada segmento. Para esta tarea, utilizamos el modelo Facebook/opt-350, que arrojó resultados superiores en comparación con el modelo Falcon. Además, optamos por la versión de 7B del modelo Facebook/opt-350 debido a las limitaciones de hardware, ya que disponíamos de dos GPUs Nvidia T4, y esta versión era el modelo más grande que podíamos acomodar.

B. Preguntas-Respuestas

Primero, implementamos la división del texto utilizando el método `CharacterTextSplitter` de la biblioteca `LangChain` [8]. Esta técnica nos permitió obtener fragmentos superpuestos del texto, con el objetivo de preservar la mayor cantidad de contexto posible en cada fragmento.

A continuación, utilizamos el modelo de embeddings `all-MiniLM-L6-v2` [1] de Hugging Face, que asigna oraciones y párrafos a un espacio vectorial denso de 384 dimensiones. Este modelo es altamente versátil y se puede utilizar en diversas tareas como la agrupación o la recuperación semántica de información.

Utilizamos la biblioteca `annoy` [2] para construir una base de datos de vectores, generando representaciones vectoriales para cada uno de nuestros fragmentos de texto. Posteriormente, empleamos esta base de datos de vectores para realizar búsquedas de similitud entre la pregunta formulada y los textos fragmentados presentes en la base de datos.

La ventaja de usar "annoy" radica en su eficiencia y efectividad para calcular rápidamente la similitud de vectores. Esto nos permite identificar los fragmentos de texto que son más similares a la pregunta formulada, proporcionando una base sólida para generar respuestas coherentes y relevantes.

Con base en los resultados de esta búsqueda de similitud, seleccionamos los tres textos más similares a la pregunta planteada. Estos textos se convierten en la base para construir nuestro prompt personalizado.

Al usar estos fragmentos relevantes en nuestro prompt, formulamos una pregunta más específica y precisa para presentar al modelo. De esta manera, podemos dirigir la atención del modelo hacia los aspectos más relevantes y obtener respuestas adecuadas y contextualmente coherentes.

VIII. CONFIGURACIÓN EXPERIMENTAL

A. Diseño del prompt

Con el objetivo de realizar un experimento, se tuvo en cuenta la notable sensibilidad del modelo a los prompts utilizados. Por lo tanto, se desarrolló un diseño cuidadoso de prompt, creando uno específico para la generación de capítulos y otro para la formulación de preguntas. Con este enfoque, pudimos observar que el modelo podía proporcionar mejores respuestas para producir información relevante para cada una de las tareas asignadas.

Prompt para preguntas

Use the following text to understand the context: [texts]
With this context answer the following question [question]

Prompt para Capítulos

Here's a section of transcription from a Youtube video. Your task is to create a title that appropriately describes the content presented in the transcription. The transcription is the following:

{text}

The best title for the previous transcription is

B. Evaluación

La evaluación de las salidas del modelo se realizó manualmente, ya que no teníamos un método cuantitativo específico para llevar a cabo la evaluación. Para abordar esta limitación, decidimos seleccionar un video que habíamos visto en su totalidad previamente. Esta elección nos permitió comparar las respuestas generadas por el modelo con nuestro conocimiento previo del contenido del video, proporcionándonos un marco de referencia para evaluar la coherencia y precisión de las respuestas. Aunque esta evaluación se basó en un enfoque cualitativo, nos proporcionó información relevante sobre el rendimiento y la capacidad del modelo para comprender y responder a preguntas específicas relacionadas con el video seleccionado.

IX. RESULTADOS

A. Capítulos

Aquí, presentaremos los resultados de los títulos generados por el modelo. Los capítulos son descripciones breves de segmentos de video que buscan capturar su contenido de manera precisa y concisa. A continuación, mostraremos los títulos generados para el video mencionado.

-
- **Prompt Length in tokens:** 289
 - **Proposed Title:** "Why Go Was So Difficult for Computers to Master"
 - **Timestamp:** 00:16:13
 - **YT Timestamp URL:** <https://youtu.be/ioV9LdMQvKs?t=973>
 - **Time Elapsed:** 1.27 seconds.
-

- **Prompt Length in tokens:** 328
 - **Proposed Title:** "Google Duplets: A Breakthrough in Natural Language Understanding"
 - **Timestamp:** 00:25:21
 - **YT Timestamp URL:** <https://youtu.be/ioV9LdMQvKs?t=1521>
 - **Time Elapsed:** 3.21 seconds.
-

- **Prompt Length in tokens:** 314
 - **Proposed Title:** "The Role of Deep Learning in Image Recognition"
 - **Timestamp:** 00:52:14
 - **YT Timestamp URL:** <https://youtu.be/ioV9LdMQvKs?t=3134>
 - **Time Elapsed:** 1.37 seconds.
-

- **Prompt Length in tokens:** 332
 - **Proposed Title:** "Collecting Millions of Images with Annotations: A Challenge Solved by Lee Feifei"
 - **Timestamp:** 00:53:41
 - **YT Timestamp URL:** <https://youtu.be/ioV9LdMQvKs?t=3221>
 - **Time Elapsed:** 2.12 seconds.
-

- **Prompt Length in tokens:** 325

- **Proposed Title:** "Using Crowdsourcing to Annotate 14 Million Images"
- **Timestamp:** 00:55:19
- **YT Timestamp URL:** <https://youtu.be/ioV9LdMQvKs?t=3319>
- **Time Elapsed:** 1.59 seconds.

X. PREGUNTAS

La selección de resultados se hizo exclusivamente en función de las preguntas planteadas, sin considerar las respuestas generadas. Esta decisión se tomó con el objetivo de evaluar la capacidad del sistema para comprender y responder adecuadamente a una variedad de preguntas, independientemente de la coherencia o precisión de las respuestas obtenidas.

Al centrarse únicamente en las preguntas, podemos evaluar la capacidad del modelo para interpretar y abordar diferentes temas y conceptos planteados por los usuarios de manera más objetiva. Este enfoque permite analizar la versatilidad y la adaptabilidad del sistema en términos de su comprensión inicial de las preguntas formuladas, sin que la calidad de las respuestas influya en la evaluación.

Sin embargo, es importante señalar que un análisis completo del sistema requeriría considerar tanto las preguntas como las respuestas generadas. Este enfoque integral permitiría evaluar no solo la comprensión inicial del modelo, sino también su capacidad para proporcionar respuestas coherentes, precisas y útiles. Al incluir las respuestas en el análisis, se podría obtener una visión más completa del desempeño del sistema, identificando áreas de fortaleza y oportunidades de mejora en la generación de respuestas.

Además, incluir las respuestas permitiría evaluar otros aspectos importantes, como la capacidad del modelo para mantener la coherencia en diálogos extendidos, la precisión en la información proporcionada y la relevancia de las respuestas en diferentes contextos. Estos factores son esenciales para una evaluación exhaustiva del rendimiento del sistema y su efectividad en aplicaciones prácticas.

| Pregunta | Respuestas |
|---|--|
| What is Deep Blue? | Deep Blue is a computer program developed by IBM to play chess. It was the first computer program to defeat a world champion in a match. |
| Who is Yann LeCun and what do they talk about him in the video? | Yann LeCun is a French-American computer scientist and artificial intelligence researcher. In the video, he talks about his work in deep learning and how it has revolutionized the field of artificial intelligence. |
| Tell me what the video is about? | The video is about a new algorithm that can generate a driving sequence without any additional information. It is a neural network based method that generates all this information by itself. It identifies what kind of movements and transformations are taking place in our driving video. |
| Ask me 3 questions that deal with the text? | What is the Turing test? What is the idea of the Turing test? What was the idea of the Turing test? What is Google Duplets? What is Google Duplets able to do? |
| What is Google Duplets? | Google Duplets is a technology developed by Google that allows a computer to understand natural language and respond to it in a natural way. It is based on deep learning and natural language processing. |
| What do they talk about in the video about Google Duplets? | Use the following text to understand the context: ["Systems for speech recognition. This is from 2015, five years ago. At that time, the war error rate before the learning was like 20 |

XI. CONCLUSIONES

A. Capítulos

En cuanto a los títulos, el modelo demuestra una notable eficiencia al resumir con precisión el contenido del fragmento de video. Sin embargo, es importante tener en cuenta que hay casos en los que los títulos generados pueden ser incorrectos o imprecisos debido a una mala transcripción o errores en la clase misma. Estos casos aislados no deben atribuirse directamente al rendimiento del modelo, sino más bien a posibles fallas en la entrada de datos o a información inadecuada proporcionada durante la clase. En general, el modelo muestra una capacidad satisfactoria para generar títulos coherentes y representativos del contenido del video, siempre que se asegure una transcripción precisa y una comunicación clara durante la clase.

B. Preguntas

La utilización del contexto proporcionado en el prompt es crucial para generar respuestas útiles y coherentes. El modelo demuestra un buen rendimiento al utilizar dicho contexto para comprender y responder adecuadamente a las preguntas planteadas. Sin embargo, es importante tener en cuenta que cuando hay errores en la transcripción o se presentan preguntas incoherentes, el modelo puede verse afectado y generar respuestas que pueden parecer incoherentes o "alucinadas". Estos casos son el resultado de la calidad de los datos de entrada y la coherencia de las preguntas planteadas, más que una limitación inherente del modelo en sí. En resumen, el uso efectivo del contexto del prompt permite obtener respuestas valiosas y coherentes, siempre que se mantenga la calidad de los datos y se planteen preguntas claras y coherentes.

XII. DISCUSIÓN

A. Limitaciones para el cuestionamiento

Una de las principales limitaciones radica en el tamaño limitado del prompt. Si queremos formular preguntas que abarquen todo el video, el modelo no es muy efectivo en responderlas, ya que solo puede considerar segmentos específicos de ese video. Esto implica que la comprensión y respuesta a preguntas que requieren un contexto más amplio pueden estar limitadas por esta restricción. Es importante tener en cuenta esta limitación al formular preguntas y esperar respuestas del modelo.

B. Restricciones de contexto

La limitación del contexto del modelo con respecto a la clase se debe a la restricción impuesta por el prompt. Esta limitación implica que el modelo no tiene acceso a un contexto completo y detallado de la clase, lo que puede resultar en imprecisiones en las respuestas generadas. Sin un contexto amplio, el modelo puede no comprender completamente el trasfondo y los detalles específicos de la clase, lo que lleva a respuestas menos precisas o incompletas. Es importante considerar esta limitación al interpretar las respuestas proporcionadas por el modelo y considerar la necesidad de

complementarlas con información adicional para obtener una comprensión más completa y precisa.

C. Limitaciones computacionales

A nivel computacional, es importante destacar que nuestra aplicación requiere el uso de una GPU (Unidad de Procesamiento Gráfico) para ejecutar nuestro modelo de manera óptima y eficiente. Esta necesidad de una GPU puede crear ciertas limitaciones, ya que no todos los dispositivos o sistemas cuentan con esta capacidad de procesamiento gráfico. Por lo tanto, es importante considerar este requisito técnico al evaluar el uso y la accesibilidad de nuestra aplicación.

XIII. TRABAJO FUTURO

A pesar de las limitaciones mencionadas, hay varios aspectos en los que se podría trabajar en el futuro para mejorar este proyecto:

- Mejora del modelo de preguntas y respuestas: Se podría explorar la posibilidad de etiquetar algunos datos para el ajuste fino con el fin de abordar las limitaciones actuales. Estos modelos podrían ser capaces de comprender y responder preguntas que requieran un contexto más amplio, superando así la restricción del prompt actual.
- Expansión del contexto disponible: Para superar las limitaciones de contexto, se podrían considerar enfoques que permitan al modelo acceder a información adicional sobre la clase. Esto podría lograrse a través de la integración de bases de datos externas, recursos en línea, o incluso la incorporación de interacciones en tiempo real con el usuario para obtener aclaraciones o detalles específicos. Al proporcionar un contexto más rico y detallado, se podría mejorar la precisión y completitud de las respuestas generadas.
- Optimización del rendimiento computacional: Dado que la aplicación actual requiere el uso de una GPU para un rendimiento óptimo, sería valioso explorar enfoques de optimización que permitan la ejecución eficiente del modelo en dispositivos sin GPU o con recursos computacionales limitados. Esto ampliaría la accesibilidad de la aplicación y facilitaría su uso en una variedad de plataformas y entornos.
- Integración de retroalimentación y corrección de respuestas: Una posible mejora sería permitir a los usuarios proporcionar retroalimentación sobre la calidad de las respuestas generadas por el modelo y, de ser necesario, corregir cualquier respuesta incorrecta. Esto permitiría una mejora continua de la precisión y confiabilidad del sistema a través de la interacción del usuario y la retroalimentación constante.

REFERENCES

- 1 Hugging Face. This is a sentence-transformers model: <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>all-MiniLM-L6-v2 .
- 2 Annoy: Approximate Nearest Neighbors Oh Yeah <https://github.com/spotify/annoy>annoy
- 3 Whisper: A Library for Generating High-Quality Speech from Text <https://github.com/openai/whisper>whisper

- 4 Falcon: A Model for Generating Instructions from Natural Language Queries <https://huggingface.co/tiiuae/falcon-7b-instruct>**falcon**
- 5 Facebook/opt-350: A Space for Exploring Text Generation with Transformers <https://huggingface.co/spaces/tloen/Facebook/opt-350/tree/main>**Facebook/opt-350**
- 6 LoRA: Low-Rank Adaptation of Large Language Models <https://github.com/microsoft/LoRA>Alora
- 7 Introducing LLaMA: A foundational, 65-billion-parameter language model <https://ai.facebook.com/blog/large-language-model-llama-meta-ai/>**llama**
- 8 Text Splitters <https://docs.langchain.com/docs/components/indexing/text-splitters>LangChain