Procesamiento de Lenguaje Natural
Proyecto Final

Rayo Mosquera, Jhon Stewar
j.rayom@uniandes.edu.co

De La Rosa Peredo, Carlos Raul
c.delarosap@uniandes.edu.co

Mario Garrido Córdoba
m.garrido10@uniandes.edu.co

Noviembre 2024

1 Introducción

El proyecto **Recuperación de Información Regulatoria y Generación de Respuestas (RIRAG)** se enfoca en desarrollar soluciones que permitan recuperar y comprender información en contextos regulatorios. Este informe detalla nuestra primera entrega, centrada en el **Subtask 1: Recuperación de Pasajes**, cuyo objetivo es mejorar el rendimiento del baseline BM25 en la recuperación de pasajes relevantes para consultas específicas.

2 Metodología

2.1 Preparación y Limpieza de Datos

2.1.1 Descripción del Conjunto de Datos

El **conjunto de datos ObliQA** es una colección de documentos regulatorios provenientes de *Abu Dhabi Global Markets (ADGM)*. Incluye 40 documentos legales y un conjunto de preguntas asociadas, distribuidas en conjuntos de entrenamiento, validación y prueba. Cada pregunta está relacionada con uno o más pasajes relevantes dentro de los documentos, proporcionando una base sólida para tareas de recuperación y generación de respuestas.

2.1.2 Preprocesamiento del Texto

Llevamos a cabo las siguientes acciones:

- **Expansión de Contracciones:** Utilizamos la librería `contractions` para expandir términos como *don't* a *do not*, evitando ambigüedades.
- **Normalización y Limpieza:** Convertimos el texto a minúsculas y eliminamos caracteres no alfanuméricos y signos de puntuación mediante expresiones regulares con la librería `re`.
- **Eliminación de Espacios Redundantes:** Eliminamos espacios en blanco adicionales y caracteres de control para mantener la consistencia en el texto.
- **Preservación de Formato Legal:** Evitamos la normalización Unicode para conservar caracteres especiales que puedan ser relevantes en contextos legales.

2.2 Tokenización, Stopwords y Stemming

Aplicamos las siguientes técnicas:

- **Tokenización:** Implementamos una función personalizada que genera **unigramas y bigramas** para capturar tanto términos individuales como combinaciones de palabras significativas.
- **Eliminación de Stopwords:** Combinamos las stopwords de `nltk` y `scikit-learn` para crear una lista adaptada al dominio, eliminando palabras comunes que no aportan significado.
- **Stemming:** Aplicamos el **algoritmo Snowball Stemmer** para reducir las palabras a sus raíces, unificando diferentes formas de una misma palabra y reduciendo la dimensionalidad.

2.3 Modelos Implementados

2.3.1 Modelos Sintácticos

BM25 Utilizamos la implementación de `rank_bm25` para construir un modelo BM25. Este modelo considera la frecuencia de términos y la longitud del documento para asignar puntuaciones de relevancia. Configuramos los parámetros $k1 = 1.5$ y $b = 0.75$, valores comúnmente utilizados que ofrecen un equilibrio entre frecuencia y longitud.

TF-IDF Implementamos un vectorizador TF-IDF utilizando `TfidfVectorizer` de `scikit-learn`. Configuramos parámetros como `ngram_range=(1,2)`, `max_features=30000`, `max_df=0.9` y `min_df=3` para optimizar la representación del texto y reducir el impacto de términos muy frecuentes o muy raros.

2.3.2 Modelo Semántico

Seleccionamos el modelo `BAAI/bge-small-en-v1.5`, un transformer preentrenado capaz de generar embeddings semánticos de alta calidad. Realizamos *fine-tuning* utilizando el conjunto de datos preparado, enfocándonos en maximizar la similitud entre pares de preguntas y pasajes relevantes. Configuramos el modelo con las siguientes especificaciones:

- **Capa de Embeddings:** Utilizamos `SentenceTransformer` con `torch_dtype=float16` para optimizar la memoria y velocidad.
- **Pooling y Normalización:** Incluimos capas de `Pooling` (utilizando el token [CLS]) y `Normalize` para mejorar la representación vectorial de los textos.
- **Función de Pérdida:** Empleamos `MultipleNegativesRankingLoss` para entrenar el modelo en tareas de recuperación de información.

2.3.3 Sistema Híbrido de Recuperación

Para combinar las ventajas de los modelos sintácticos y semánticos, desarrollamos un sistema híbrido que integra ambos enfoques mediante un promedio ponderado de sus puntuaciones:

$$\text{Puntuación Híbrida} = \alpha \times \text{Puntuación Semántica} + (1 - \alpha) \times \text{Puntuación Sintáctica}$$

Donde $\alpha = 0.65$ asigna mayor peso al componente semántico. Normalizamos las puntuaciones de ambos modelos entre 0 y 1 para asegurar una combinación equilibrada.

2.4 Entrenamiento y Fine-Tuning

Para entrenar el modelo semántico:

- **Configuración de Entrenamiento:**
  - **Épocas:** 10.
  - **Batch size:** 64 con `gradient_accumulation_steps=4` para simular un batch de 256.
  - **Learning rate:** 2×10^{-5}.
- **Estrategias de Optimización:** Utilizamos `warmup_ratio=0.1` y desactivamos `gradient_checkpointing` para mejorar la eficiencia.
- **Evaluación Continua:** Empleamos `InformationRetrievalEvaluator` para monitorizar el rendimiento en cada época, enfocándonos en métricas clave.
- **Implementación de Hard Negatives:** Incorporamos ejemplos negativos difíciles extraídos con BM25 para robustecer el entrenamiento.

3 Experimentos y Resultados

3.1 Configuración Experimental

Realizamos los experimentos utilizando los conjuntos de datos de entrenamiento, validación y prueba proporcionados por ObliQA. Las implementaciones se llevaron a cabo en Python, utilizando librerías como `sentence-transformers`, `nltk` y `scikit-learn`. Los modelos fueron entrenados y evaluados en una GPU NVIDIA A40 para acelerar el procesamiento.

3.2 Evaluación y Métricas

Utilizamos las métricas **Recall@10**, **MAP@10**, **Recall@20** y **MAP@20** para evaluar el rendimiento de los modelos. Estas métricas son estándar en tareas de recuperación de información y proporcionan una visión integral de la capacidad de los modelos para recuperar y ordenar pasajes relevantes. La evaluación se realizó con la herramienta `trec_eval`, siguiendo los estándares de la comunidad en IR (Information Retrieval).

3.3 Rendimiento Comparativo

Modelo	**Recall@10**	**MAP@10**	**Recall@20**	**MAP@20**
BM25 (Baseline)	0.7791	0.6415	0.8204	0.6453
Modelo Semántico	0.8103	0.6286	0.8622	0.6334
Sistema Híbrido	**0.8333**	**0.7016**	**0.8704**	**0.7053**

Tabla 1: Rendimiento comparativo de los modelos en el conjunto de prueba.

Los resultados muestran que el sistema híbrido supera significativamente al baseline BM25 y al modelo semántico individual en todas las métricas evaluadas.

4 Discusión

Los experimentos confirman que la combinación de modelos sintácticos y semánticos mejora la recuperación de información en el dominio regulatorio. El modelo BM25 es eficaz en recuperar pasajes con coincidencias exactas, pero carece de capacidad para entender sinónimos o términos relacionados. Por otro lado, el modelo semántico captura relaciones profundas entre términos, pero puede perder precisión en coincidencias exactas.

El sistema híbrido equilibra estas limitaciones, logrando un mayor **Recall@10** y **MAP@10**. La elección de $\alpha = 0.65$ refleja la importancia de la semántica en este dominio, sin descuidar la sintaxis.

Además, observamos que el modelo semántico ajustado mantiene capacidades generales del modelo base, lo que es beneficioso para manejar consultas fuera del dominio específico.

5 Conclusiones

Hemos desarrollado un sistema de recuperación de pasajes que supera al baseline BM25 mediante la integración de modelos sintácticos y semánticos. Nuestro enfoque híbrido demuestra ser efectivo en capturar tanto coincidencias exactas como relaciones semánticas, lo que es crucial en el contexto regulatorio.

Los resultados obtenidos validan nuestra hipótesis y resaltan la importancia de combinar técnicas tradicionales y avanzadas en NLP. Este trabajo sienta las bases para abordar el Subtask 2, enfocado en la generación de respuestas.