Extracción de entidades en sentencias judiciales usando LLaMA-2

Francisco Vargas[1,2], Alejandro González Coene[1,2], Gaston Escalante[1,2], Exequiel Lobón[1,2], and Manuel Pulido[1,3]

[1] FaCENA, Universidad Nacional del Nordeste, Corrientes
[2] Legalhub S. A., Buenos Aires
[3] Instituto de Modelado e Innovación Tecnológica, CONICET
franciscopvargas98@gmail.com

Resumen La extracción de información de accidentes viales disponible en sentencias judiciales es de relevancia para la cuantificación de costos de las aseguradoras. La extracción de entidades tales como porcentajes de incapacidad física y/o psicológica y montos involucrados es un proceso difícil aun para expertos por las sutiles argumentaciones en las sentencias. Se propone un procedimiento que se divide en dos pasos, la segmentación de la sentencia e identificación del segmento relevante y luego la extracción de las entidades. Se comparan dos metodologías, un método clásico basado en expresiones regulares. La segunda metodología esta basada en la división del documento en bloques de n-tokens para luego vectorizarlos con modelos multi-lenguajes para búsquedas semánticas (*text-embedding-ada-002/MiniLM-L12-v2*). Posteriormente se aplican LLMs (LLaMA-2 7b, 70b y GPT4) con *prompting* al bloque relevante para la extracción. En el caso de LLaMA-2 se realiza un sintonizado fino (*finetuning*) con LoRA. LLaMA-2 7b aun con temperatura nula presenta un significativo número de alucinaciones en las extracciones que disminuye sustancialmente con el sintonizado. El rendimiento de la metodología basada en el vectorizado de los segmentos y el posterior uso de los LLMs supera ampliamente al método clásico. La exactitud del método clásico es 39,5 %, la de LLaMA-2 70b base 61,7 % y con sintonizado 79,4 %, mientras que para GPT-4 Turbo es 86,1 %.

Keywords: Reconocimiento de entidades nombradas · Grandes modelos de lenguaje · Textos legales.

1. Introducción

Los procesos judiciales en la Argentina están compuestos por diversas etapas de comunicación entre los actores, demandados y entes judiciales, que finalizan en una sentencia. Nos centraremos en los documentos del fuero civil relacionados con las demandas por daños y perjuicios. Para cuantificar el daño de cada lesión sobre el actor, un perito cuenta con un conjunto reglamentado de porcentajes de incapacidad que se deben asignar a cada lesión que el actor presente, la incapacidad provocada sobre el actor puede ser de dos tipos, física o psicológica. En la actualidad no existe un registro de los valores porcentuales que terminan siendo asignados. Además cada uno de estos valores asignados tiene asociado un monto indemnizatorio, no prefijado. El análisis de datos puede ayudar a identificar tendencias en la relación entre el monto de la indemnización y la incapacidad considerando que la asignación de los jueces a la causa puede ser aleatoria. Tanto el demandante como el defensor pueden utilizar estas tendencias para solicitar o apelar indemnizaciones con una mayor probabilidad de ser aprobadas. Para la estimación de tendencias es necesario el reconocimiento automático para poder procesar gran cantidad de sentencias del fuero. Esto tiene el gran desafío que dichas entidades a reconocer son escasas y el texto es extenso, muchas veces entidades similares se pueden encontrar en contextos diferentes como por ejemplo citas de otros casos que no tienen relación con el fallo actual y esto interfiere con la calidad de la extracción.

Para reducir las interferencias en la extracción es indispensable segmentar el texto y seleccionar los segmentos relevantes aprovechando una característica de las sentencias que es el patrón de su estructura, esto nos permite utilizar una segmentación jerárquica a priori sin la necesidad de un modelo [1]. Una condición indispensable al momento de seleccionar el segmento es la existencia de símbolos o palabras claves. Esta regla basada en una palabra clave resultan ser simple pero no determinante al momento de la selección de los segmentos candidatos. Por lo tanto, para mejorar la identificación de segmentos potenciales de contener algún tipo de incapacidad se propone y evalúa una búsqueda por similitud basada en la vectorización de los segmentos y la búsqueda de consultas.

La extracción de entidades nombradas (NER) es muy común en la ciencia de datos y ampliamente utilizada en el ámbito legal [15] aunque la problemática comúnmente atacada dista de la presentada en este artículo ya que el reconocimiento de entidades generalmente supone que la frecuencia de aparición de una entidad (por ejemplo, el nombre de una persona, juez o actor) es baja dentro de un documento y aún más baja entre documentos [8], nuestro caso al tratar de reconocer valores porcentuales y montos dentro del texto implica que estos pueden aparecer repetidas veces en un mismo documento, estar presentes en todos los documentos y representar diferentes magnitudes en cada caso donde la única manera de diferenciar estas entidades es a partir del contexto.

Los últimos avances en el procesamiento de lenguaje natural han revolucionado el entendimiento de textos mediante el uso de grandes modelos de lengua-

je[4]como GPT y LLaMA [12]. Estos nos permiten mediante el *prompt engineering* optimizar el comportamiento de un modelo de lenguaje sin necesidad de un *fine-tuning* especifico para la tarea. En entornos productivos es necesario tomar una decisión teniendo en cuenta el costo beneficio del uso de un modelo sintonizado frente a uno no sintonizado así como también es importante tener en cuenta las ventajas del uso de modelos abiertos como LLaMA-2 frente a modelos cerrados como GPT-3 y GPT-4 que son actualizados con el tiempo y cuyos datos de entrenamiento son desconocidos [2].

En este trabajo proponemos un camino para resolver la identificación de las entidades en sentencias. Se propone un procedimiento de dos etapas. En la primera se identifican los segmentos mas relevantes del texto. Para este fin se evalúan diferentes segmentadores de textos, basados en expresiones regulares y vectorización (Sección 2.2). En la segunda etapa se utilizan grandes modelos de lenguaje, GTP-3.5, LLaMA-2 y GPT4, para la extracción de entidades y una comparativa con métodos basados en palabras clave con expresiones regulares (Sección 2.4). Los resultados de la aplicación de estas metodologías y la comparación de sus rendimientos se muestran en Sección 3. La aplicación de conocimientos previos sobre los textos para llevar a cabo la extracción es también evaluada.

2. Metodologías

2.1. Base de datos

Los documentos utilizados en este trabajo son sentencias judiciales de primera instancia relacionadas a demandas por daños y perjuicios del fuero civil. Estos documentos están disponibles en la plataforma del Poder Judicial de la Nación Argentina (PJN)[10] en formato digitalizado, son públicos y de libre acceso.

Los documentos fueron recopilados utilizando un *scraper* del sitio web del PJN. Dentro del sitio se seleccionan aquellas sentencias relacionadas con accidentes. La base de datos así generada contiene un conjunto de 650 sentencias judiciales de primera instancia, las cuales sus sentencias firmes fueron dictadas en el mes de Octubre de 2023.

Se realiza antes del análisis un preprocesamiento y limpieza de los datos que consiste en una serie de pasos. En primer lugar, se clasifican los documentos según su formato, separando los PDFs con textos extraíbles de los PDFs que consisten únicamente en imágenes o escaneados, descartando los que se encuentran en la segunda categoría. Luego se utiliza la herramienta PyPDF2 [13] para extraer el texto de estos documentos y se utilizan expresiones regulares para eliminar cualquier información irrelevante o no deseada de los mismos. Entre éstas, se elimina los códigos de las sentencias que se repiten en el cabezal de cada unas de las páginas.

Como último paso se realiza un segundo filtrado a nivel de documento para separar las sentencias que se encuentren dentro del alcance definido, siendo este accidentes de tránsito incapacitantes sin muertes. Este filtro consistió en tener las

[4] También conocidos como Large Language Models o LLMs en inglés

carátulas de cada sentencia y, con conocimiento dentro del dominio, se definieron palabras claves que debían estar o no para que cada sentencia se encuentre dentro del campo de estudio definido. Utilizando estas expresiones regulares se clasifica cada carátula y filtran las sentencias que están en el alcance de este estudio.

Al pasar la base de datos de 650 sentencias por los filtros anteriores se obtienen 278 sentencias dentro del alcance de este trabajo. Para la generación de la base de datos con etiquetas se utilizó un procedimiento híbrido, en el cual primero se utiliza un modelo de lenguaje para la extracción de etiquetas y a posteriori se usa la intervención humana para el control de calidad. Este se describe en Sección 2.5.

2.2. Segmentación de Documentos

Método de Segmentación con RegEx La segmentación con expresiones regulares (RegEx) se puede utilizar para dividir las sentencias judiciales en secciones más pequeñas y manejables para los modelos de lenguaje. El objetivo es identificar segmentos de texto dentro de las sentencias judiciales que contengan información sobre incapacidades físicas o psicológicas, así como el porcentaje de incapacidad y montos asociados. Se resume las entidades de interés en la tabla 1.

Entidad	Propiedades de la entidad
Incapacidad Física	Porcentaje de incapacidad | Monto indemnizatorio
Incapacidad Psicológica	Porcentaje de incapacidad | Monto indemnizatorio
Monto Daño Moral	Monto indemnizatorio

Cuadro 1. Entidades para la extracción.

El proceso de segmentación con RegEx elegido esta compuesto por:

- **Buscar el signo de '%' dentro del texto:** El propósito de esta búsqueda es encontrar posibles indicadores de porcentaje de incapacidad en el texto. Expresión regular: `r"[\w\d\s\n,.]{0,1}%"`
- **Determinación del contexto:** Para cada coincidencia de porcentaje, se establece un contexto alrededor de la misma. Este contexto se define mediante una ventana de 500 caracteres antes y después de la posición de la coincidencia, lo cual genera un contexto de aproximadamente 320 tokens.

Método de Segmentación por Vectorización y Búsqueda Semántica: La segmentación de documentos por búsqueda vectorial es un proceso por el cual se representa el contenido del texto en un espacio vectorial utilizando un modelo del tipo transformer [11]. A partir de este "vectorizador"se representa a cada uno de los segmentos que componen el documento por un conjunto limitado de vectores. Dentro de este espacio vectorial, podemos llevar a cabo la búsqueda

por comparación utilizando la similitud cosenoidal mediante la librería FAISS [5]. En este trabajo se utilizó el modelo de embeddings paraphrase-multilingual-MiniLM-L12-v2 de Huggingface [11]. Este modelo es particularmente relevante debido a su capacidad para manejar varios idiomas específicamente el español, su eficiencia y su capacidad de vectorizar a nivel de tokens.

El proceso de segmentación se realiza mediante los siguiente pasos:

- **División de texto en bloques:** Se divide el texto plano del documento en segmentos de tamaño fijo de 120 tokens. Se optó por este tamaño por las limitaciones del modelo de embeddings [11] (máximo de 124 tokens) utilizado y para facilitar la extracción de características sin comprometer la exactitud semántica.
- **Vectorizado de bloques de texto:** Estos bloques de texto son vectorizados con el modelo de embedding y almacenados utilizando la librería FAISS [5] (*Facebook AI Similarity Search*), la cual genera índices específicos para optimizar la búsqueda.
- **Búsqueda vectorial:** Se realiza una búsqueda vectorial sobre los vectores generados a partir de una consulta la cual será comparada con cada uno de los bloques de textos vectorizados.
- **Generación de la consulta para la búsqueda:** Para determinar el texto de la consulta para realizar la búsqueda por similitud cosenoidal se realizó un Tf-idf (*Term frequency – Inverse document frequency*) sobre los bloques de textos esperados. De esta manera se obtienen palabras claves a utilizar en la búsqueda, para así obtener una mayor exactitud.
- **Contexto de bloques de texto:** A partir de los bloques resultantes de la búsqueda semántica se realiza una expansión de dichos bloques, concatenando con los bloques anteriores y posteriores generando nuevos vectores a partir de estos 3 bloques (bloque resultantes, más el anterior y el posterior) para mejorar el contexto de dichos bloques de texto. Dando así un bloque de texto de 360 tokens seleccionado para la extracción.

2.3. Extracción de las entidades por expresiones regulares

El método clásico utilizado para la extracción de entidades es el uso de expresiones regulares. Este enfoque, es el que esta implementado en los sistemas actuales por lo que es usado como la referencia de performance base para medir el impacto que tiene el uso de modelos de lenguaje. Para el funcionamiento de este método de extracción se requiere definir a priori un solo segmento relevante, con éste se realiza:

- **Búsqueda de palabras clave para identificar el tipo de incapacidad:** Dentro del contexto, se utilizan expresiones regulares para buscar palabras claves que indiquen el tipo de incapacidad asociada al porcentaje encontrado. Esto incluye palabras como 'física' o 'psicológica'. Dependiendo de las palabras clave encontradas, se determina el tipo de incapacidad asociada al porcentaje.

- **Extracción del porcentaje de incapacidad:** Además de identificar el tipo de incapacidad, se utilizan expresiones regulares para extraer el porcentaje de incapacidad del contexto. Esto se logra buscando números seguidos de un signo de porcentaje dentro del contexto.
  La expresión regular es: r"(\d+(?:,\d+)?(?:.\d+)?)\s*%"
- **Extracción del monto:** De la misma manera se extraen los montos a partir de la identificación del signo pesos y números dentro del segmento relevante.

2.4. Modelos de lenguaje para la extracción de entidades

En los últimos años se han desarrollado varios grandes modelos de lenguaje y existen numerosos esfuerzos para tal fin. Para este trabajo se optó por utilizar los modelos de la familia LLaMA-2[14] de Meta, considerando que es uno de los modelos mas populares de la comunidad open source y que se ajusta a las necesidades para este trabajo. Es un modelo estado del arte, y si bien hay algunos posteriores (e.g. Mixtral), este sentó las bases para muchos otros. Por otro lado a los efectos de comparación del rendimiento se utilizaron los modelos GPT-3.5 y GPT-4.

Metodología general El proceso general para la extracción de entidades con modelo de lenguaje utiliza una metodología *Retrieval-Augmented Generation* (RAG) como se resume en la Figura 1. Dados el documento, un conjunto de *querys* y el *prompt* optimizados para el mismo, en primer lugar, se utiliza el segmentador con las *querys* para dividir el texto del documento en diferentes bloques, generando sus *embeddings* y almacenándolos. En una segunda etapa, el modelo de lenguaje es alimentado con cada bloque, y sus vectores correspondientes, y con un *prompt* optimizado para extraer las entidades y sus propiedades.

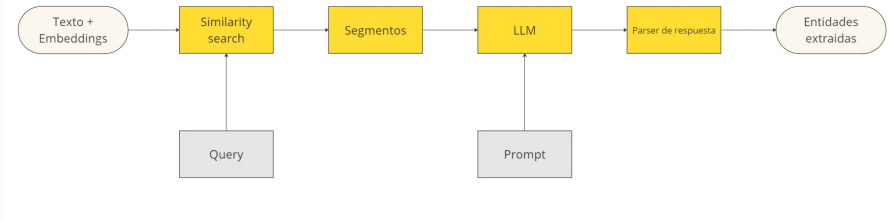

Figura 1. Pipeline de extracción con LLM.

2.5. *Dataset* de entrenamiento

Se dispone de 278 documentos filtrados y preprocesados (ver Sección 2.1), los cuales son útiles para entrenar el modelo. Para llevar a cabo el entrenamiento

del modelo, se necesita una base de datos que contenga los segmentos relevantes de cada documento etiquetados con las entidades a extraer. Para el proceso de etiquetado de la base, se propuso utilizar un LLM con intervención humana posterior.

En primer lugar nos concentramos en la obtención de un prompt óptimo para la extracción de entidades. Nos referimos por *prompt* a la instrucción en lenguaje natural que recibe el modelo. Para esta optimización existen diferentes técnicas de *prompting* (e.g. pensar paso por paso[9], seguir formatos específicos para facilitar el entendimiento y la estructura del texto, entre otras), que son utilizadas para armar instrucciones eficientes y lograr las mejores respuestas de los grandes modelos de lenguaje. Los prompts utilizados pueden ser encontrados en [5].

El modelo utilizado para este etiquetado inicial de la base de datos fue GPT-4 de OpenAI. El etiquetado consiste en utilizar una RAG, en la cual se extraen los segmentos del documento, tal y como se describe en la sección 2.2, usándolo como entrada al LLM junto con el *prompt* para que de esta manera el LLM extraiga las entidades a partir de la información relevante brindada. En una segunda etapa, a las etiquetas resultantes obtenidas con el LLM se les realiza un control de calidad manual para verificar que efectivamente son las entidades del documento. Este etiquetado mixto nos permite tener un ahorro considerable de horas hombre al momento de realizar las correcciones de los datos, ya que solo se deben corregir extracciones mal realizadas por el modelo.

La base de datos etiquetada luego es utilizada con dos propósitos, una parte de esta se usa para el *finetuning*. La otra se reserva para tener una referencia (conjunto de prueba) para evaluar el rendimiento de los diferentes LLMs en los distintos experimentos que realizamos.

El segmentador utilizado para el etiquetado de la base de datos corresponde al modelo de *embeddings* "*text-embedding-ada-002*"de OpenAI. Teniendo estos segmentos, se alimenta al modelo de lenguaje que extrae las entidades. Una vez procesados todos los documentos, como se mencionó, se procedió a realizar una depuración manual de los datos sobre el total de los expedientes. Ésta consistió en revisar cada uno de los PDFs y las entidades extraídas, que son las incapacidades físicas, psicológicas, psicofísicas con sus características (monto y porcentaje) y el monto indenmizatorio por daño moral corrigiendo las etiquetas si se lo requiere.

A partir de este proceso manual, se obtienen las etiquetas correctas que el modelo debe proporcionar para cada extracción. De esta manera, se logra el etiquetado del conjunto de datos completo, del cual quedaron un total de 1120 muestras, correspondientes a las 4 entidades en los 278 documentos. Este conjunto de muestras se denomina *Dataset* 1. Asumimos la hipótesis de que la muestra de una entidad contribuirá a que el modelo entrenado pueda reconocer otras del mismo tipo o similares en textos nuevos.

Las muestras del conjunto de datos corregido manualmente todavía no son perfectas para realizar el entrenamiento. Existen etiquetas que serían imposibles de extraer ya que no se encuentran en los segmentos proporcionados por el

[5] http://github.com/Fran-98/JAIIO-ASAID-141

segmentador. En estos casos la etiqueta debería estar vacía. Entonces, se generó un segundo conjunto de datos. Se realizaron búsquedas manuales de las entidades dentro de los segmentos pertinentes. Si las entidades no fueron encontradas en el segmento, esas muestras fueron descartadas. También se podría haber optado por utilizar esas muestras y etiquetarlas como vacías; para el armado de este *dataset* se decidió descartarlas. Después de esta limpieza, quedaron 861 muestras, y este conjunto se denomina *Dataset* 2.

Como base de datos de prueba se utilizaron 30 documentos dentro del mismo dominio que resultan en 120 muestras. Estos documentos no pertenecen al conjunto de entrenamiento y su preparación fue similar a la depuración manual realizada para el *Dataset* 2 de entrenamiento.

2.6. Sintonizado fino

Debido a la restricción de VRAM de 96Gb disponible (correspondientes a 2 Nvidia A40), para los entrenamientos se optó por usar técnicas de cuantización. Este es el proceso por el cual se reduce la precisión de los pesos del modelo para que ocupen menos espacio en memoria y se requieran menos recursos para realizar el entrenamiento y la inferencia. La cuantización también aumenta la velocidad de inferencia. En este trabajo se utilizó la librería *bitsandbytes*[3] para realizar la cuantización del modelo de 32 bits de punto flotante a 8 bits entero, i.e. FP32 $\rightarrow$ int8. La degradación de la de los pesos no afecta la exactitud de la inferencia significativamente y permite utilizar modelos de mayor número de parámetros, e.g. 70b, obteniendo de esta manera mejores resultados que si se opta, por ejemplo, por un modelo de 13b sin ningún tipo de cuantificación [7].

Por esta restricción se optó por utilizar el método de *finetuning Quantized Low Rank Adaptation*[4], QLoRA, el cual es un método derivado de [6] al que se le adiciona la cuantización. El LoRA introduce la posibilidad de entrenar adaptadores que poseen muchos menos parámetros que los modelos base y que pueden ser integrados a éstos para lograr un *finetuning* efectivo utilizando muchos menos recursos que si intentáramos actualizar todos los pesos del modelo. Estos métodos pertenecen a la familia de métodos PEFT (*Parameter-Efficient Fine-Tuning*) [17]. QLoRA en este caso nos permitió realizar el *finetuning* en los recursos computacionales disponibles.

Se realizaron 2 entrenamientos, uno por cada *dataset*, dejando un 10 % de cada conjunto de datos como conjunto de validación. Las capas entrenadas fueron las de atención, con un *learning rate* de 5e-5, durante 3 épocas y rango de LoRA de 8. De éstos se seleccionó el *checkpoint* mas adecuado para cada *dataset*, buscando el punto óptimo en el cual el modelo haya generalizado la información de entrenamiento pero no presente un sobre ajuste.

3. Resultados

3.1. Segmentación

Durante la fase de segmentación de documentos, es esencial evaluar la eficacia de la búsqueda semántica y su capacidad para identificar correctamente los seg-

mentos necesarios para la extracción de datos por parte del modelo de lenguaje. Sin esta información, el modelo no podría extraer las entidades requeridas, ya que no tendría acceso a los datos necesarios. Se llevó a cabo una evaluación de calidad (QA) de la segmentación en las sentencias. Esta evaluación permitió determinar si la búsqueda semántica efectivamente recupera los segmentos con los datos correctos para su posterior extracción. El resultado de este QA, realizado sobre el conjunto de prueba de 30 sentencias, mostró un porcentaje de exactitud del 83,33 % con el modelo "*sentence-transformer/paraphrase-multilingual-MiniLM-L12-v2*"[11] y una exactitud del 80,91 % con el modelo de embeddings "*text-embedding-ada-002*"de OpenAI.

3.2. Resultado de las extracciones

Los resultados obtenidos utilizando los segmentos del modelo *text-embedding-ada-002* con el LLaMA-2 7b y el LLaMA-2 70b sintonizados se muestran en Figura 2a).

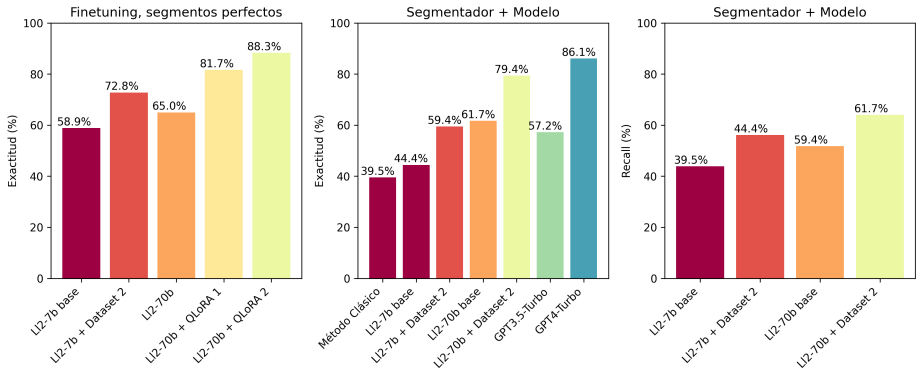

Figura 2. a) Impacto del *finetuning* en la exactitud de cada modelo considerando un segmentador perfecto. b) Exactitud del procedimiento completo considerando el segmentador y el modelo. c) Resultados de recall del procedimiento completo considerando el segmentador y los modelos.

Se obtuvieron mejoras apreciables de los rendimientos obtenidos cuando se sintoniza con los *datasets*. Por otro lado hay una ganancia de exactitud de 9,4 % entre el *Dataset* 1 al *Dataset* 2. Este último logra superar ampliamente al primero, incluso teniendo menor cantidad de muestras. La única diferencia, además de la cantidad de muestras, es la limpieza realizada, por lo que se aprecia la gran importancia de tener un *dataset* de calidad a la hora de entrenar cualquier modelo. El mejor resultado, LLaMA-2 70b sintonizado con el *Dataset* 2, tiene una mejora del 17,7 % frente al modelo base. El sintonizado en LLaMA-2 7b tiene un menor impacto con una diferencia de 13,9 %.

Las exactitudes mostradas en la Figura 2a) corresponden a un segmentador perfecto; sin embargo, al realizar la inferencia de forma automática, es probable que el segmentador seleccione algunos segmentos de manera incorrecta. La eficacia del proceso completo, incluyendo el segmentador y el modelo, se presenta en la Figura 2 panel b). Se observa el *recall* calculado para los modelos sintonizados, figura 2 panel c), este es mas bajo que la exactitud medida por lo que el modelo es mas conservador a la hora entregar un valor e intenta mantener una alta fiabilidad a la hora de acusar una extracción. Esto último nos permite estar seguros de sus resultados, pero a su vez, se pierde una substancial parte de la información que se podría estar aprovechando.

Los resultados obtenidos con el método propuesto superan ampliamente al obtenido con el método clásico, RegEx. Este método da un rendimiento de 39.5 %. mientras el LLaMA-2 70b con *finetuning* da un rendimiento de 79.4 %. El rendimiento es significativamente superior a GTP3-5 pero inferior a GPT4 que obtiene 86.1 %.

Por otro lado, se evidencia un fuerte deterioro del rendimiento de 19.4 % en el caso del modelo LLaMA-2 7b base cuando los segmentos no son perfectos comparado a cuando los segmentos son perfectos (Figura 2). En estas situaciones, cuando no se identifica correctamente la entidad en el segmento, el modelo tiende a generar respuestas no válidas. Esto se debe a que cuando el modelo no encuentra la entidad correcta en el segmento tiende a alucinar una respuesta inexistente en el segmento. Este resultado motivó la necesidad de realizar una evaluación de las alucinaciones en los modelos.

3.3. Alucinaciones

Las alucinaciones son un gran problema al utilizar LLMs, ya que debido a su naturaleza estos modelos generan respuestas sin importar si estas son correctas o no. El caso de extracción de entidades no es ajeno a este problema y puede ocurrir que las extracciones se vean afectadas por este afán de encontrar una respuesta.

Para cuantificar este fenómeno, se creó un conjunto de datos con los segmentos descartados para formar el *Dataset* 2, donde no están presentes las entidades a extraer. Se utilizó una cantidad de segmentos equivalente a los que se obtendrían de 30 sentencias judiciales. Los resultados de este experimento se muestran en la Figura 3. El modelo de LLaMA2-7b tiene un rendimiento muy deficiente cuando la información no está presente dentro del segmento. Un resultado destacable es la mejora significativa con el ajuste fino del modelo LLaMA2-7b, con una diferencia de 47,78 % frente a su versión base. Esta diferencia es mucho menor cuando se compara al caso del modelo LLaMA2 70b. En este caso el modelo base tiene una baja cantidad de alucinaciones ya que estos tienen una mejor capacidad para seguir las instrucciones y comprender mejor los textos, con una diferencia 7,78 %. En ambos casos, la mejora del ajuste fino es notable para reducir las alucinaciones.

Para disminuir las alucinaciones y su impacto, se evaluó un método de detección de alucinaciones usando las probabilidades asociadas a los *tokens* generados

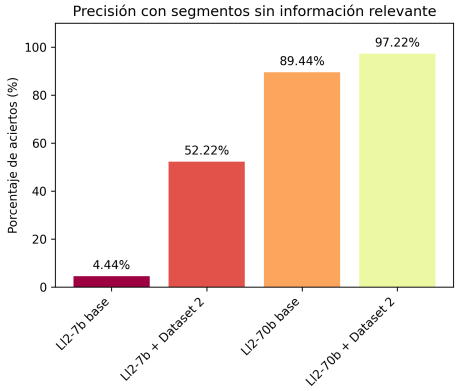

Figura 3. Análisis de los modelos con segmentos que no contienen información.

por el modelo. El método de mitigación se basa en que la probabilidad de un token es una cuantificación de la certeza que posee el modelo sobre la generación del *token* seleccionado como siguiente. Esta probabilidad o certeza viene condicionada por el *prompt* y el contexto del texto ya generado.

Si tomamos a las probabilidades como $p_1, p_2, ..., p_n$, según [16] el método mas adecuado para detectar una alucinación es tomando la mínima probabilidad dentro del conjunto, $p_{min} = \text{mín}(p_1, p_2, ..., p_n)$. Podemos definir un umbral mínimo de probabilidad, $p_{min} < p_u$ de esta manera toda generación que posea en su conjunto de probabilidades una probabilidad menor a este será considerada una potencial alucinación.

Se analizó el método del umbral mínimo para la detección de alucinaciones con el *dataset* de prueba, teniendo en cuenta tanto las extracciones exitosas como las que no lo fueron. Utilizando el umbral mínimo como método de detección, no fue posible reconocer algún patrón o valor para el umbral, se encuentran casos en los que el modelo tiene un porcentaje muy bajo pero la extracción es correcta, como también ocurre lo contrario. Entonces vemos claramente que un LLM puede estar muy seguro de una respuesta incorrecta. Como conclusión se determina que para el caso de extracción de entidades este método por si solo es insuficiente para detectar extracciones incorrectas.

3.4. Estadística de las extracciones

Realizadas la extracciones se pasa a realizar la estadística de las entidades extraídas. Esta información que es de utilidad para las aseguradoras como también para el demandante y el defensor quienes pueden utilizar esta información para solicitar o apelar indemnizaciones. Una de las cantidades estimadas a partir de las entidades extraídas es el valor punto, VP. Éste es un cálculo usado para estimar el monto asignado por los jueces por cada punto porcentual de incapacidad de un individuo. Estas estadísticas son cruciales para que las aseguradoras

estimen la liquidez necesaria para cubrir juicios como también para solicitar o apelar indemnizaciones. Para determinar el VP consideramos los montos atribuidos a la incapacidad psicológica IP_m y su porcentaje IP_p, los montos de la incapacidad física IF_m y su porcentaje IF_p y el monto atribuido al daño moral DM_m, resultando

$$VP = \frac{IP_m}{IP_p} + \frac{IF_m + DM_m}{IF_p}, \tag{1}$$

donde en el último término para el monto del daño moral se esta asumiendo que el porcentaje de la incapacidad física es mas determinante y significativo que el porcentaje de la incapacidad psicológica.

Se procedió a realizar la inferencia en todas las sentencias del año 2023 correspondientes a las jurisdicciones circunscriptas a Capital Federal. A partir de estas sentencias se extrajeron las entidades en forma automática siguiendo el flujo RAG descripto en Figura 1 con el modelo de mejor rendimiento, GPT-4, y a partir de éstas se estimó el valor punto usando (1). En términos generales se puede observar en Figura 4 en función del mes de la sentencia que el valor punto tiene una correlación significativa con el valor acumulado siguiendo el índice de precios al consumidor (IPC); sin embargo también existen comportamientos particulares que no pueden ser descriptos a través del IPC.

De esta manera el cálculo del valor punto puede ser obtenido para todo un año con un procedimiento automático que requiere de una media hora de tiempo de cómputo en promedio. El procedimiento tradicional para obtener el valor punto consistía en analizar manualmente cada sentencia para extraer cada entidad y luego realizar el cálculo manualmente, lo cual llevaba mas de una semana de trabajo, por mes, para un abogado experimentado en el tema.

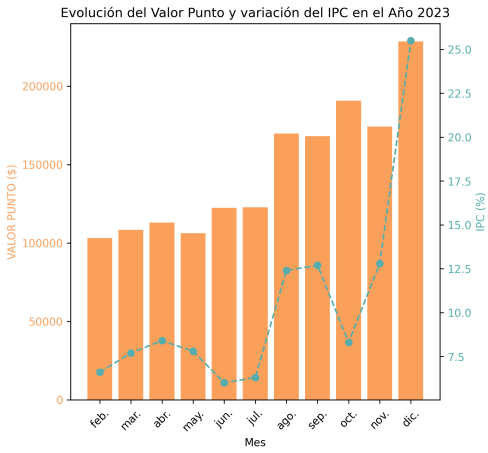

Figura 4. Variación del valor punto e IPC durante 2023, obtenido a través del procedimiento descripto en Figura 1

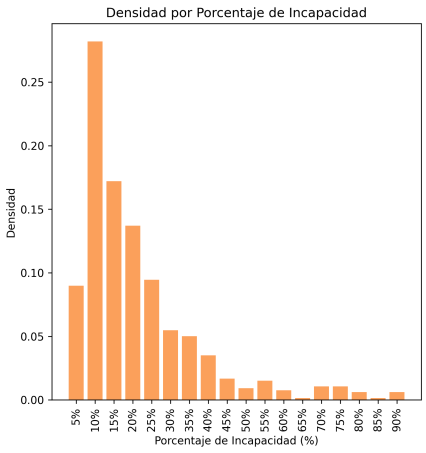

Figura 5. Distribución de porcentajes de incapacidad por sentencia obtenido a partir de la extracción de entidades.

Otra información que es de relevancia para aseguradoras y para la defensa es la distribución de los porcentajes de incapacidad. Es decir si consideramos todas las sentencias como la muestra, usando el porcentaje de incapacidad extraído de cada sentencia se puede obtener su histograma. Considerando la distribución de los porcentajes de incapacidades, el valor punto y el número de causas, éstos nos permiten realizar estimaciones estadísticas de los costos involucrados. De esta manera, las aseguradoras pueden estimar la liquidez necesaria para cubrir juicios. La distribución de los porcentajes de incapacidad en las sentencias obtenida para las sentencias de 2023 se muestra en la Figura 5.

Tomando el mes de Octubre, que es el mes para el cual se tiene disponible la extracción manual completa, podemos analizar que error nos introduce la extracción automatizada. Comparando el valor calculado con los datos reales obtenidos de la depuración manual y el valor calculado con los datos extraídos automáticamente, obtenemos un error porcentual relativo del 12,9 %.

4. Conclusiones

El método basado en modelos de lenguaje planteado en este trabajo demuestra una mejora significativa frente a los métodos clásicos basados en expresiones regulares para la extracción de entidades en sentencias judiciales. A su vez, mostramos que un LLM *open source* puede mejorar significativamente esta tarea siendo entrenado con una base de datos relativamente pequeña obteniendo una exactitud máxima de 88.3 %, superando a modelos comerciales bien establecidos como GPT-3.5 y siendo comparable a GPT-4, aunque este es superior 86.1 %. El impacto del *finetuning* en LLaMA-2 70b fue de 29.4 %, se obtuvo un 58.90 % en el modelo base comparado a un porcentaje de 79.4 % obtenido con el modelo entrenado. El sintonizado fino permitió disminuir significativamente la cantidad de alucinaciones encontradas en el modelo LlaMA-2 7b base.

Los sistemas actuales en uso para la identificación de incapacidades están basados en el uso de expresiones regulares, por lo que en este trabajo se enfocó en medir el impacto del uso de modelos de lenguaje en comparación con el método de expresiones regulares. Obteniéndose una mejora sustancial con el uso de grandes modelos de lenguaje (GPT-4 y LlaMA-2). El análisis no incluyó la diferencia de rendimiento entre los grandes modelos de lenguaje LLM con modelos de lenguaje mas clásicos como BERT.

La técnica propuesta brinda la posibilidad de poder reemplazar el trabajo manual de extracción de datos a partir de sentencias judiciales de accidentes por un flujo de trabajo automático, segmentador/LLM, lo que permite poder analizar un gran volumen de documentos, pudiendo así, obtener información que antes era inviable, entre las que cabe citar la distribución de probabilidades de incapacidades y el valor punto mensual, como se demuestra en Sección 3.4.

El procedimiento descripto en este trabajo permite la discriminación del valor punto estimado por jurisdicciones e incluso a nivel de juez por lo que el análisis estadístico general realizado podría extenderse a la detección de anomalías jurisdiccionales o a nivel de juez en el valor punto y/o en los porcentajes de

incapacidades. Por otro lado potencialmente se podrían detectar cambios en la cola de la distribución de incapacidades por jurisdicción o por juez.

Reconocimientos Los entrenamientos y las inferencias con los grandes modelos de lenguajes fueron realizadas en los servidores de cómputo del CECONEA (UNNE).

Referencias

1. Bayomi, M., Lawless, S.: C-hts: A concept-based hierarchical text segmentation approach (may 2018)
2. Chen, L., Zaharia, M., Zou, J.: How is chatgpt's behavior changing over time? (2023), arXiv 2307.09009
3. Dettmers, T.: bitsandbytes. https://github.com/TimDettmers/bitsandbytes (2021)
4. Dettmers, T., Pagnoni, A., Holtzman, A., Zettlemoyer, L.: Qlora: Efficient finetuning of quantized llms (2023), arXiv 2305.14314
5. Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P.E., Lomeli, M., Hosseini, L., Jégou, H.: The faiss library (2024), arXiv 2401.08281
6. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models (2021), arXiv 2106.09685
7. Jin, R., Du, J., Huang, W., Liu, W., Luan, J., Wang, B., Xiong, D.: A comprehensive evaluation of quantization strategies for large language models (2024), arXiv 2402.16775
8. Karaa, W.B.A.: Named entity recognition using web document corpus. MatSciRN: Other Electronic (2011), https://api.semanticscholar.org/CorpusID:9633141
9. Kojima, T., Gu, S.S., Reid, M., Matsuo, Y., Iwasawa, Y.: Large language models are zero-shot reasoners (2023), arXiv 2205.11916
10. Poder judicial de la nación argentina, https://www.pjn.gov.ar/
11. Reimers, N., Gurevych, I.: Código fuente paraphrase-multilingual-minilm-l12-v2 huggingface., https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2
12. Roumeliotis, K.I., Tselikas, N.D., Nasiopoulos, D.K.: Llms in e-commerce: A comparative analysis of gpt and llama models in product review evaluation. Natural Language Processing Journal **6**, 100056 (2024). https://doi.org/10.1016/j.nlp.2024.100056
13. Thoma, M.: Pypdf2, https://pypi.org/project/PyPDF2/
14. Touvron, H., Martin, L., Stone, K., et al.: Llama 2: Open foundation and fine-tuned chat models (2023), arXiv 2307.09288
15. Vardhan, H., Surana, N., Tripathy, B.K.: Named-entity recognition for legal documents. In: Hassanien, A.E., Bhatnagar, R., Darwish, A. (eds.) Advanced Machine Learning Technologies and Applications. pp. 469–479. Springer Singapore, Singapore (2021)
16. Varshney, N., Yao, W., Zhang, H., Chen, J., Yu, D.: A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation (2023), arXiv 2307.03987
17. Xu, L., Xie, H., Qin, S.Z.J., Tao, X., Wang, F.L.: Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment (2023), arXiv 2312.12148