CLASIFICACIÓN DE ÁRBOLES DE PROBLEMAS PARA LA FORMULACIÓN DE PROYECTOS SOCIALES

Juan Pablo Díaz Pardo & Andrea Prada Quintero

Directores: Juan Pablo Pájaro Hernández & Juan Pablo Mora López

Maestría en Analítica para la Inteligencia de Negocios

Facultad de Ingeniería

Pontificia Universidad JAVERIANA Bogotá

2021

CONTENIDO

I. ENTENDIMIENTO DEL NEGOCIO

Inversiones Gutiérrez García (IGG) es una empresa de consultoría en el sector de telecomunicaciones, tecnología e innovación con más de 30 años en el mercado. Como parte de su misión, la empresa busca ser un aliado estratégico en todos los temas relacionados con el desarrollo de soluciones tecnológicas en *Big Data*, visión por computador, desarrollo de *software*, redes neuronales, entre otras. La empresa cuenta con cuatro unidades de negocio: ISIPOINT (gestión y administración de zonas de estacionamiento públicas o privadas), MIRECARGA (plataforma transaccional propia para comercialización de recargas y venta de productos de telecomunicaciones), CONTACTO MÓVIL (prestación de servicios y contenidos en el mundo digital) y GENERACIÓN MÓVIL (portafolio de productos, servicios y soluciones de tecnología y telecomunicaciones). El proyecto desarrollado hace parte de esta última línea de negocio.

1.1. Objetivos de negocio

Para entender la relación que tiene el proyecto con los objetivos de negocio de IGG, es necesario describir los pilares de la estrategia organizacional a través de su misión y visión:

MISIÓN: "Ser una empresa innovadora en el sector de telecomunicaciones, que responda a las necesidades del mercado corporativo y empresarial, por medio de la integración de la tecnología y de conceptos como inteligencia artificial, *Deep Learning* y modelos de aprendizaje. Logrando el posicionamiento de la marca en el país y en la región, siendo LÍDERES en soluciones integrales de tecnología con alta calidad al cliente final."

VISIÓN: "Ser el aliado tecnológico de empresas nacionales e internacionales en el sector de las TICs y la movilidad, para el desarrollo de soluciones innovadoras, con tecnología de punta, que integren nuevos modelos de aprendizaje y soportadas por sistemas seguros y confiables, alineados con iniciativas globales como la de *Smart Cities."* (Inversiones Gutiérrez García, 2021)

De acuerdo con la misión y visión descritas anteriormente, es posible observar que IGG actualmente se encuentra enfocada en el desarrollo de soluciones empresariales apalancadas en modelos de aprendizaje automático y profundo. Teniendo esto en cuenta, el proyecto analítico deberá estar alineado con este propósito, y en este caso en particular, lo que se busca desarrollar es un modelo de aprendizaje automático que facilite la construcción de árboles de problemas dentro la metodología del marco lógico (MML).

La MML es una herramienta para facilitar el proceso de conceptualización, diseño, ejecución y evaluación de proyectos (Ortegón, Pacheco , & Prieto, 2005). Fue diseñada originalmente como respuesta a tres problemas comunes:

- Planificación de proyectos carentes de precisión.
- Proyectos que no tienen una ejecución exitosa.
- Falta de como luciría un proyecto si fuese exitoso.

Además de encarar de la mejor forma estas problemáticas, también provee algunas ventajas como: **1)** aportar una terminología uniforme mejorando así la comunicación, **2)** tener un formato para llegar a acuerdos precisos, suministrar un temario analítico, **3)** enfoque en los aspectos críticos, **4)** obtención de información para organizar y preparar en forma lógica el plan de ejecución (además de lo necesario para su ejecución, monitoreo y evaluación) y finalmente **5)** proporcionar una estructura única con la información relevante.

A grandes rasgos, la MML incorpora cuatro elementos analíticos que son la guía de este proceso: análisis del problema, análisis de involucrados, jerarquía de objetivos y selección de una estrategia de implementación óptima, este trabajo se enfocará únicamente en el análisis de problemas y sus efectos/causas directas.

Dentro del análisis de problemas, el primer paso consiste en identificar la problemática que se desea intervenir, así como sus causas y efectos. En general, el proceso requiere: **1)** identificar y analizar los problemas principales, **2)** establecer cuál es el problema central teniendo en cuenta criterios de prioridad y selectividad, **3)** definir los efectos y causas más importantes, **4)** construir el árbol de problemas y revisar la validez de este. Por ejemplo, en la ilustración 1 se evidencia el análisis de una empresa de transporte, cuya problemática central es la alta accidentalidad de sus automotores.

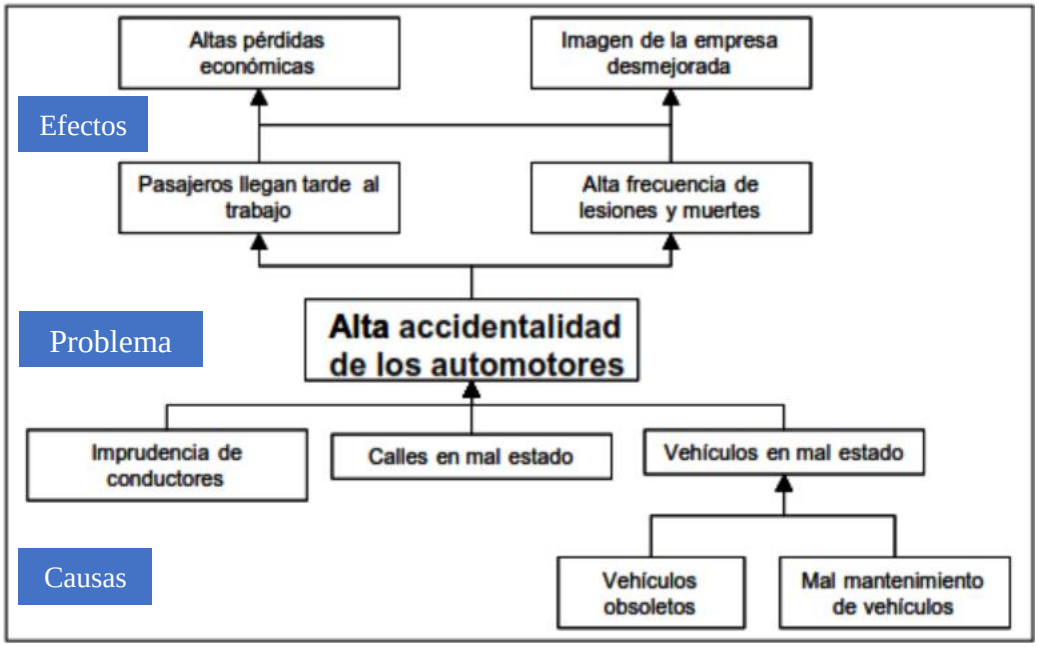


Ilustración 1. Ejemplo de un árbol de problema

De acuerdo con los manuales de la MML (Ortegón, Pacheco , & Prieto, 2005), se debe tener en cuenta que el problema central debe formularse con una connotación negativa, el análisis de causa/efecto se debe hacer con base en la problemática central escogida y no se debe confundir el problema con la ausencia de la solución. Es importante analizar los nodos críticos y realizar una matriz de incidencias.

IGG actualmente está trabajando en un proyecto de analítica *non-profit*, el cual busca encontrar una mejor manera de validar árboles de problemas sociales usando algoritmos de aprendizaje automático e inteligencia artificial (IA). En general, en Colombia es muy poco lo que se ha avanzado en temas de IA para mejorar el diseño de políticas públicas, por lo cual esta tarea es relevante dado que el proceso de validación es difícil y costoso. Esto ocurre, debido a que se requieren profesionales de múltiples disciplinas y existen elementos transdisciplinarios/ sistémicos, así como intereses particulares. (Pájaro Hernández, 2018).

En un escenario ideal y con un proceso más eficiente, donde un algoritmo de aprendizaje automático sea capaz de validar oportunamente los árboles de problemas, se tendrían dos beneficios principales: **1)** evitar la malversación de recursos públicos a causa de proyectos mal formulados y **2)** optimizar el tiempo y los recursos humanos dentro del proceso de aprobación de

los proyectos. Todo lo anterior tendrá un impacto positivo, al mejorar la calidad de vida de poblaciones vulnerables.

Para realizar la validación de un árbol de problemas previamente formulado, se proponen tres alternativas: **1)** sugerir cuál es el problema central y sus respectivas causas/efectos, **2)** calcular el grado de asociación entre diadas de oraciones (problema y causa/efecto) desde un punto de vista semántico y **3)** sugerir conceptos que no se tuvieron en cuenta dentro del contexto para dar mayor profundidad al entendimiento del problema a través de una ontología ya existente. Lo que se pretende en este caso, es tener una medida objetiva que permita ayudar a "centrar el análisis de causas y efectos entorno a un solo problema central. Lo que permite acotar el análisis y ser más efectivo en recomendar soluciones" (Pájaro Hernández, 2018) Teniendo en cuenta lo anterior, las metodologías basadas en procesamiento de lenguaje natural (PLN) resultan idóneas para abordar las necesidades de negocio.

"El PLN es el campo de la ciencia computacional y lingüística que se ocupa de las interacciones entre los computadores y el lenguaje humano" (Treleaven, 2014). En este campo, se encuentran varias aplicaciones de negocio, entre las cuales encontramos: análisis de sentimientos, traducción automática, reconocimiento de entidades nombradas (NER[1]), modelamiento de tópicos, clasificación de textos, entre muchas otras.

En trabajos similares de modelos analíticos aplicados a la formulación de árboles de problemas, se tiene como punto de partida la investigación realizada por (Pájaro Hernández, 2018). En este trabajo se realizó un servicio web basado en PLN para la validación de árboles de problemas a través del cálculo de la similitud semántica entre parejas de oraciones usando como principal recurso léxico el *WordNet* (Miller, 1995). El *pipeline* de trabajo utilizado por este autor fue: la tokenización de palabras (*Tokenizer*), partición de oraciones (*Sentences Breaking*), etiquetado gramatical (*Part of Speech Tagging*) y NER.

Partiendo de esta primera aproximación del PLN aplicada a la validación de árboles de problemas, se busca responder las siguientes preguntas: ¿es posible usar métodos de PLN más robustos para este propósito? ¿cuáles podrían ser los modelos para utilizar? ¿vale la pena su implementación desde el punto de vista de las partes interesadas?

[1] *Named entity recognition*, por sus siglas en inglés.

1.2. Objetivos de la minería de datos

Dentro de los métodos de aprendizaje supervisado de texto, existen una gran variedad de alternativas para representar los textos en un formato numérico comprensible para los algoritmos de IA, entre los cuales se destacan: n-gramas de palabras, n-gramas de caracteres, vectores de palabras (*word embeddings*) y modelos basados en lenguaje (*transformer*). Los dos primeros se basan en representaciones por frecuencias para diferentes combinaciones de palabras o caracteres, mientras que los últimos dos, consideran características más complejas del lenguaje humano para construir representaciones vectoriales de palabras u oraciones. Los modelos basados en lenguaje y particularmente los modelos *transformer* conformados por redes neuronales profundas (Liu, 2019) (Yang, 2019), han demostrado ser superiores a otros métodos en varias tareas como análisis de sentimientos, búsqueda de respuestas e inferencia del lenguaje natural (Wang, 2018). Estos modelos *transformer* han permitido avances importantes dentro de las tareas del PLN, pues permiten apalancar casos de uso específicos con modelos de gran escala previamente entrenados y transferencia de aprendizaje (Terechshenko, 2021).

De acuerdo con los objetivos de negocio planteados y el alcance que tienen los modelos basados en PLN, se proponen tres objetivos relacionados con el ejercicio de clasificación y minería de textos. La estrategia escogida para abordar el proyecto está basada en la metodología CRISP-DM (Wirth, 2000), la cual es idónea para el desarrollo de proyectos analíticos en la actualidad.

1) **Clasificación:** Desarrollar un modelo de aprendizaje automático para la validación del problema, causas y efectos de un árbol de problemas previamente formulado. Dicho modelo deberá clasificar un conjunto de oraciones en alguno de estos tres grupos, con el fin de sugerir una posible representación de la problemática planteada.
2) **Minería de texto**:
    - Desarrollar un modelo para el NER que permita en trabajos futuros vincular las entidades identificadas a una ontología ya existente.
    - Desarrollar un modelo que permita calcular el grado de similitud semántica entre el problema central y cada una de sus díadas.

Dentro de las métricas correspondientes para medir el desempeño en la clasificación de oraciones y el NER, se tienen los siguientes indicadores tradicionales de un ejercicio de clasificación:

Tabla 1. Métricas de desempeño[2]

$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$	$Precision = \frac{TP}{TP + FP}$
$Recall = \frac{TP}{TP + FN}$	$F1 - Score = 2 \times \left(\frac{Precision \times Recall}{Precision + Recall}\right)$
$AUC = \frac{Recall}{\frac{FP}{TP + FP}}$	$Micro\ Average = \sum_{c=0}^{3} \frac{x_c \times n_c}{n}$

1.3. Pregunta analítica, objetivos y *pipeline* general

Teniendo en cuenta el alcance del proyecto, a continuación, se plantea la pregunta analítica, el objetivo general y los objetivos específicos.

Pregunta analítica: ¿Es posible validar la estructura de un árbol de problemas a través de la aplicación de modelos robustos de PLN?

General: Desarrollar modelos robustos de PLN para la validación de árboles de problemas dentro de la MML.

Específicos:

- Desarrollar un modelo de aprendizaje automático para la clasificación de oraciones en un árbol de problemas.
- Desarrollar un modelo para el NER que permita en trabajos futuros vincular las entidades identificadas a una ontología ya existente.
- Desarrollar un modelo que permita calcular el grado de similitud semántica entre el problema central y sus respectivas díadas.

[2] Donde TP: número de verdaderos positivos, TN: número de verdaderos negativos, FP: número de falsos positivos y FN: número de falsos negativos, x: precision, recall o f1-score, c: clase, n: tamaño de la muestra.

Pipeline general

A continuación, se muestra el *pipeline* general del proyecto, no obstante, dada la cantidad de tareas específicas en cada caso, se hará un *pipeline* específico en cada sección con el fin de dar claridad de los pasos realizados.

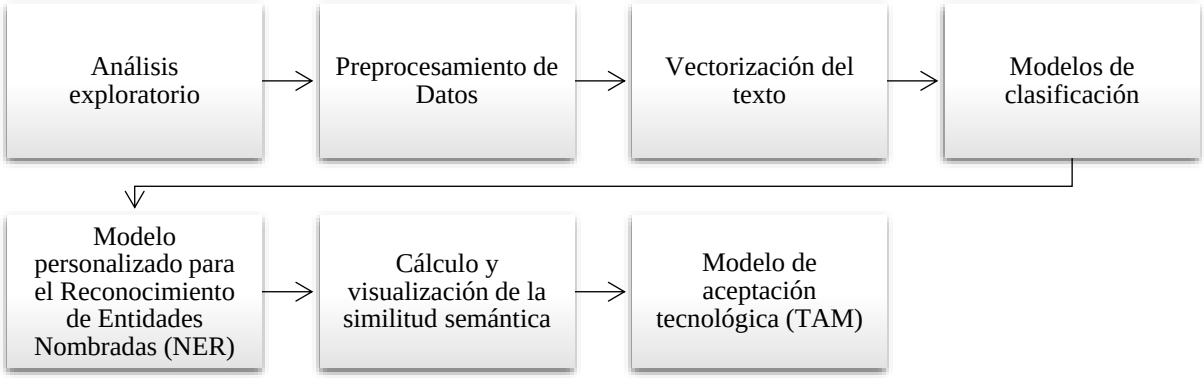


Ilustración 2. Pipeline general del proyecto

II. ENTENDIMIENTO DE LOS DATOS

2.1. Recolección inicial de los datos

El set de datos proporcionado por IGG cuenta con 1.583 proyectos de inversión en formato .csv, los cuales fueron aprobados entre el 2018 y 2019 por parte del Departamento Nacional de Planeación (DNP). Este ente gubernamental es el encargado de aprobar diversos proyectos sociales sometidos en el formato de la MML, así como el seguimiento de políticas públicas en Colombia.

2.2. Descripción de los datos

La base de datos de proyectos se encuentra etiquetada según el tipo de oración[3] (problema, causa o efecto), teniendo en total 13.037 oraciones. También se cuenta con un código interno de la organización, el cual no aporta información para el análisis y por ende no será tenido en cuenta.

[3] Dentro del archivo no existe distinción entre los niveles de causa/efecto.

2.3. Exploración de los datos

Para poder hacer la exploración de los datos fue necesario realizar una serie de transformaciones y preprocesamiento de las oraciones que se explicará con mayor detalle en la sección de preparación de los datos.

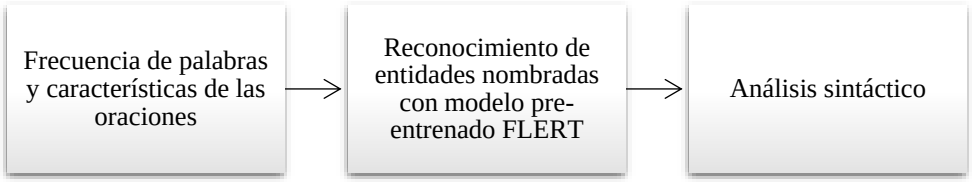


Ilustración 3. Pipeline para la exploración de datos

Frecuencia de palabras y características de las oraciones: Con el fin de entender cuáles son las palabras más frecuentes en el corpus, se realizó esta nube de palabras como primera aproximación al análisis exploratorio de las oraciones.

Ilustración 4. Mapa de las palabras de todos los proyectos

En la ilustración anterior, se evidencia que los proyectos tratan temas de transporte y la necesidad latente de mejoras en cuanto a: tiempos, costos, mantenimiento de vías e intercomunicación de zonas rurales y urbanas.

Cuando se explora la cantidad de efectos y causas por problema, se encuentra que la norma, (a excepción de algunos proyectos) es tener la misma cantidad de efectos y causas, lo cual hace sentido ya que normalmente se tiene un efecto por cada causa. Por lo tanto, una posible hipótesis al respecto es que, en la construcción de estos árboles, se tienen elementos redundantes, que hacen que está equivalencia 1:1 no se de. Posteriormente, en la gran mayoría de casos se cuenta con un

promedio de 3 a 5 efectos/causas, con algunas excepciones en las cuales puede llegar a extenderse a 25.

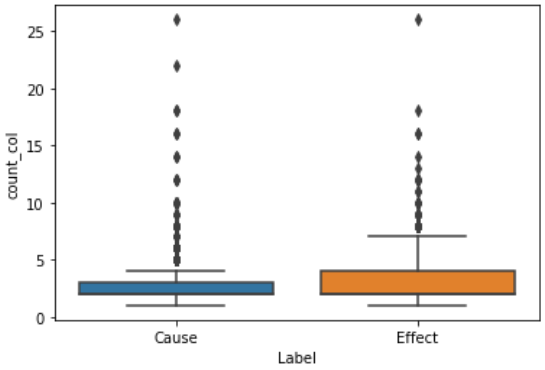


Ilustración 5. Cantidad de efectos/causas por problema

También es posible observar que dentro de la redacción de un problema existe una tendencia a extenderse más frente a las causas o efectos. Normalmente el problema cuenta con un máximo de 40 palabras y un promedio de 18, aunque hay algunos casos atípicos que se extienden hasta las 250. En cuanto a los efectos/causas se evidencia una longitud máxima de 20 palabras con un promedio de 6-8 palabras, donde la extensión máxima puede llegar hasta las 50 palabras.

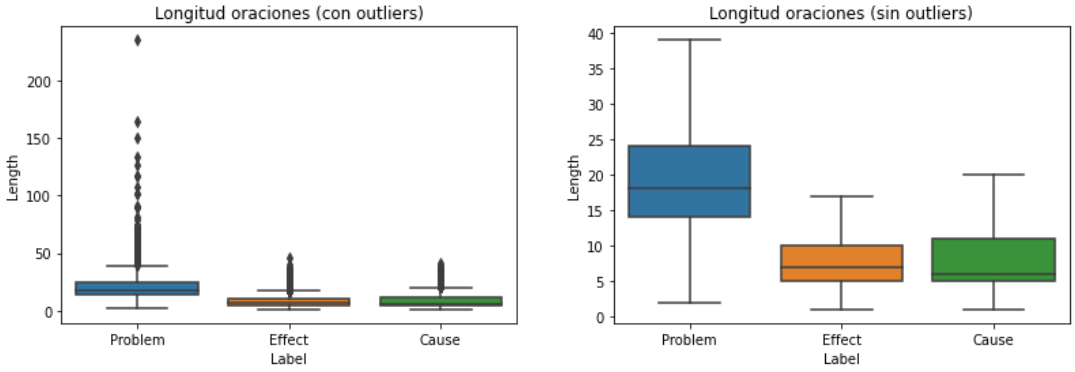


Ilustración 6. Longitud de problemas, causas y efectos.

Reconocimiento de entidades nombradas (NER): Con el fin de explorar las entidades presentes en el corpus, se realizó una nube de palabras con todos los entes identificados utilizando el modelo pre entrenado FLERT (Schweter, Flert: Document-level features for named entity recognition., 2020). Las etiquetas identificadas por este modelo pre-entrenado son: ORG (Organizaciones), MISC (Otros), PER (Personas) y LOC (Locaciones), respectivamente.

Tabla 2. Entidades nombradas modelo FLERT

A partir de la tabla anterior, se concluye que este modelo pre-entrenado base reconoce de manera adecuada la mayoría de las locaciones pertenecientes a municipios de Cundinamarca, no obstante, vemos algunas excepciones como el municipio de Chaguaní o varias veredas/vías del departamento (Vía Vereda la María, Vereda San Miguel, Vía Siberia – Tenjo, Vía San Vicente), las cuales fueron etiquetadas como “otros”. Otras entidades nombradas relevantes hacen referencia a temas de violencia y educación, por ejemplo: “PVCA” (personas víctimas del conflicto armado) y “Colegio San Jorge” e “Institución Carlos Giraldo”, las cuales fueron clasificadas como organizaciones. Se sabe por el contexto de estos proyectos que existen varias instituciones educativas en el corpus, no obstante, estas no fueron identificadas. Como otros temas relevantes en el corpus se encuentran temas legislativos y normativos (otros): “Ley 142”, “Ley 1276”, “ISO 9001”, “Código de policía”, etc. Por último, al consolidar todas las entidades nombradas del corpus, se hace evidente que las entidades de locaciones dominan por frecuencia sobre las demás etiquetas:

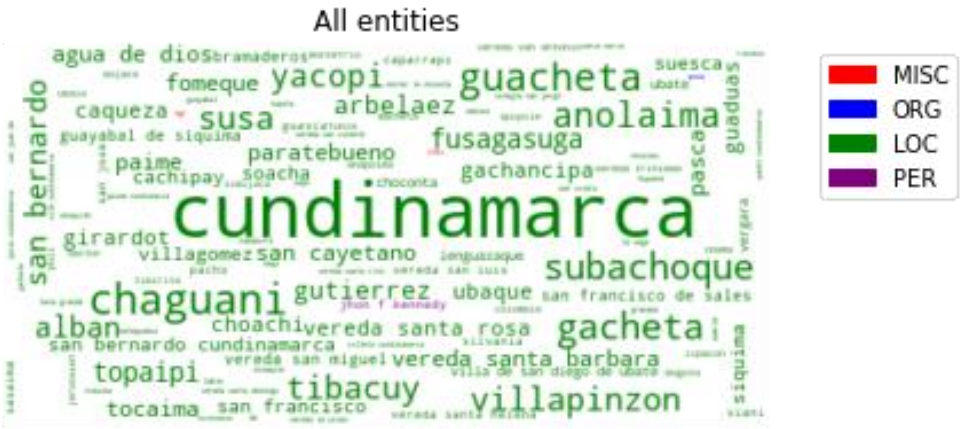


Ilustración 7. Mapa de las entidades de todos los proyectos

En general, vemos que este modelo pre-entrenado es un muy buen punto de partida para el modelo de NER que se quiere implementar, no obstante, si un futuro se quieren vincular estas entidades nombradas a una ontología, es necesario robustecer este proceso a través de un etiquetado manual y un nuevo entrenamiento para sobre ajustar este modelo base, tal como la describe la documentación del repositorio de GitHub (Schweter, flairNLP / flair, 2021). Este proceso se discutirá con mayor detalle en la sección de NER.

Análisis sintáctico de las oraciones: Como parte del proceso de análisis exploratorio se realizó un breve análisis sintáctico de las oraciones del árbol de problemas, con el fin de entender si existen diferentes relaciones entre las palabras dependiendo de la oración. La siguiente figura muestra la proporción de los diferentes patrones sintácticos identificados en cada grupo de oraciones:

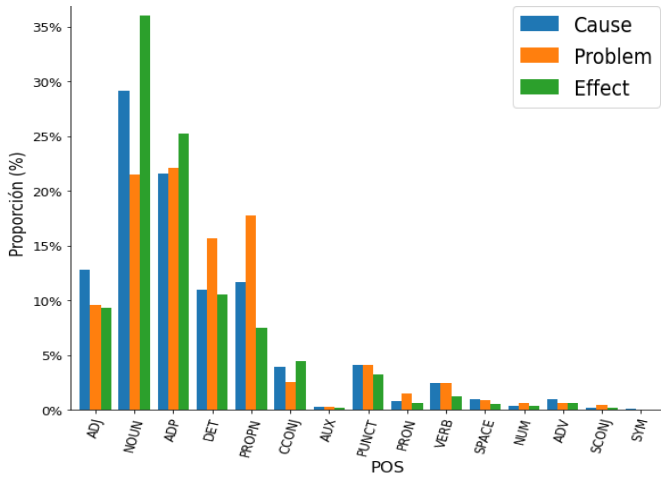


Ilustración 8. Proporción part-of-speech (POS) por tipo de oración

Tabla 3. Descripción y ejemplos del etiquetado part-of-speech

Etiqueta	Descripción	Ejemplos
ADJ	Adjetivo	"salado", "grande", "juvenil", "limpio","áspero"
NOUN	Sustantivo	"crema", "café", "estrella", "navaja", "martillo", "libros"
ADP	Adposición	"María, **la hija de mi amiga**, es muy lista" "Rafael, **el joven de la panadería**, me invitó a pasear hoy"
DET	Determinante	"el", "la", "lo", "los", y "las"
PROPN	Sustantivo propio	"Diego", "Chile", "Nestlé", "Madrid"
CCONJ	Conjunción coordinante	"y", "ni", "o", "o sea"
AUX	Verbo auxiliar	"haber", "empezar", "ir", "deber", "acabar", "comenzar", "llevar"
PUNCT	Puntuación	".", ",", "(", ")", "?", "!", "¿"
VERB	Verbo	"cantar", "bailar", "estudiar", "jugar", "dormir"
SPACE	Espacio	" "
NUM	Número	"1", "2017", "dos", "sesenta", "IV", "X"
ADV	Adverbio	"aquí", "ahí", "allí", "ahora", "luego", "después", "bien", "mal", "así", "mucho", "poco", "bastante", "sí", "también", "no", "tampoco", "quizás", "acaso"
SCONJ	Conjunción subordinante	"que", "aunque", "pero"
SYM	Símbolo	"$", "%", "§", "©", "+", "–", "×", "÷", "="

Tal como se puede observar en la ilustración anterior, efectivamente existen patrones sintácticos interesantes dependiendo del tipo de oración. Por ejemplo, nótese que los “problemas” tienen una mayor proporción de sustantivos propios (PROPN) y determinantes (DET), las “causas” una mayor cantidad de adjetivos (ADJ) y los “efectos” una mayor cantidad de sustantivos (NOUN). A nivel general, vemos que la formulación de árboles de problemas carece del uso de verbos (VERB), verbos auxiliares (AUX) y adverbios (ADV) y, al contrario, tiene una alta cantidad de adposiciones (ADP).

2.4. Verificación de la calidad de los datos

Al ser problemas sociales previamente analizados y aprobados por expertos, se considera un corpus adecuado para la realización de este análisis. Por otra parte, se encuentran únicamente 3 proyectos sin oraciones, es decir existen 3 árboles en blanco que se eliminan por ser minoría y no tener un impacto significativo en la base de datos general.

III. PREPARACIÓN DE LOS DATOS

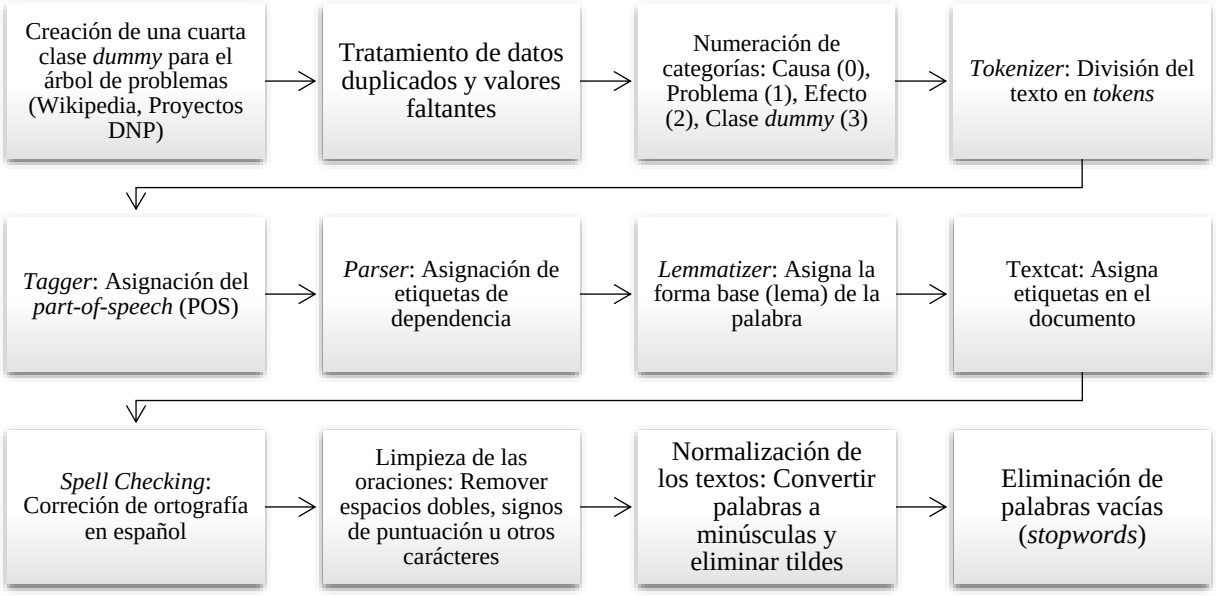


Ilustración 9. Pipeline para la preparación de los datos

Creación de una cuarta clase *dummy* para el árbol de problemas: El objetivo del proyecto es la validación de árboles de problemas, por lo que es necesario crear una nueva clase que no corresponda a una causa, problema o efecto, y que está sirva como una categoría exógena que absorba todo el ruido que no corresponda a un árbol de problemas. Para hacer esto, la propuesta planteada fue nutrir el *set* de datos original con oraciones externas que se asemejaran de cierta manera a la estructura de las oraciones originales y su dominio de palabras (entidades nombradas de Cundinamarca, temas sociales, culturales, económicos, etc.). Por este motivo, para la construcción de esta cuarta clase se utilizaron dos fuentes externas: **a)** las entradas en Wikipedia pertenecientes a los municipios de Cundinamarca y **b)** un listado de proyectos reales del DNP disponible en la página de datos abiertos (Departamento Nacional de Planeación, 2021).

a) **Entradas de Wikipedia:** Se creo un algoritmo de *web-scraping* para realizar la recolección rápida de oraciones para las entradas correspondientes de cada municipio. A continuación, se muestra como ejemplo arbitrario los posibles conjuntos de oraciones sintéticas que surgen para el municipio de Bojacá (Wikipedia, 2021), el cual es una entidad nombrada real en el conjunto de datos original.

Tabla 4. Aproximación propuesta para la generación de oraciones sintéticas a partir de Wikipedia

Oración original	Posibles oraciones sintéticas
"La infraestructura física del las instituciones educativas rurales y urbanas del municipio de bojaca, presentan fallas de tipo físico que requieren intervención inmediata para la garantizar la calidad del servicio."	• Bojacá es uno de los 116 municipios del departamento de Cundinamarca, Colombia. • Se encuentra ubicado en la Provincia de Sabana Occidente, a 40 km de Bogotá. • La población fue fundada en 1573. Tiene una histórica tradición religiosa que ha convertido su templo y la imagen de su Virgen en un sitio de peregrinación. • Hace parte del Área Metropolitana de Bogotá según el censo del DANE en 2005.

b) **Proyectos del DNP:** Este repositorio incluye más de 349.000 proyectos ejecutados por diversas entidades gubernamentales dentro del territorio nacional en temas de: educación, transporte, deporte y recreación, vivienda, agricultura y desarrollo rural, etc. De este conjunto se filtraron los proyectos correspondientes a Cundinamarca en las categorías de: Agricultura y desarrollo rural, Ambiente y desarrollo sostenible, Ciencia, tecnología e innovación, Comercio, industria y turismo, Cultura, Deporte y recreación, Educación, Empleo público, Minas y energía, Planeación, Relaciones exteriores, Salud y protección social, Tecnologías de la información y las comunicaciones, Trabajo, Transporte, y Vivienda, ciudad y territorio. El número de oraciones resultantes después de aplicar este filtro fue de 16.868, sin embargo, para efectos prácticos sólo un subconjunto aleatorio de 1.600 oraciones fue introducido al modelo con el fin de no inducir demasiado ruido al modelo. A continuación, se muestra como ejemplo ilustrativo algunas oraciones arbitrarias (sin preprocesamiento) de este repositorio, las cuales servirán como insumo para el modelo de clasificación:

Tabla 5. Ejemplos de oraciones para el repositorio de Datos Abiertos

	Ejemplos oraciones
1)	CONSTRUCCION DEL POLIDEPORTIVO CUBIERTO LA GAITANA EN EL CASCO URBANO DEL MUNICIPIO DE GUACHETA CUNDINAMARCA
2)	ESTUDIOS Y DISEÑOS PARA LA CONSTRUCCION DE UNA INSTITUCION EDUCATIVA EN EL MUNICIPIO DE SESQUILE CUNDINAMARCA
3)	ADQUISICION DOTACION PARA LA AMPLIACION DE SERVICIOS DE RADIOLOGIA E IMAGENES DIAGNOSTICAS DE LA ESE HOSPITAL NUESTRA SENORA DE LAS MERCEDES DE FUNZA CUNDINAMARCA
4)	ADECUACION DEL ESPACIO PUBLICO Y OBRAS DE URBANISMO FASE II DEL PARQUE PRINCIPAL DEL MUNICIPIO DE VIOTA CUNDINAMARCA
5)	APOYO PARA LA PROMOCION Y ACCESO EFECTIVO A PROCESOS CULTURALES Y ARTISTICOS EN EL MUNICIPIO DE LA CALERA
6)	MANTENIMIENTO Y EMBELLECIMIENTO PARA EL BUEN USO DEL ESPACIO PUBLICO EN EL MUNICIPIO DE VILLETA CUNDINAMARCA
7)	DESARROLLO Y ACCESO INTEGRAL A LA VIVIENDA DIGNA EN CONDICIONES DE HABITABILIDAD PARA LA POBLACION VULNERABLE DEL MUNICIPIO DE TOCANCIPA

Al sumar las oraciones de ambas fuentes, se obtuvo un total de 2.870 oraciones que fueron incorporadas dentro de la fase de modelamiento como una cuarta clase.

Preprocesamiento para el PLN: En esta etapa se realizó todo el preprocesamiento estándar de PLN para entrenar los modelos de clasificación de aprendizaje automático. En primer lugar, se realizó el tratamiento de oraciones duplicadas y valores faltantes, donde se eliminaron oraciones vacías o duplicadas dentro de un mismo árbol de problemas, con el fin de no incorporar información vacía o redundante. Se realizó la numeración de las categorías, donde Causa (0), Problema (1), Efecto (2) y la clase *dummy* (3).

La gran mayoría del análisis léxico, morfológico y sintáctico de las oraciones como Tokenización, *POS Tagging*, Lematización, etc. se hizo con la ayuda de la librería *Spacy* (anexo 1) y el *pipeline* en español[4] (Explosion, 2021) el cual utiliza el corpus *AnCora* (Taulé, 2008), *WikiNER* (Nothman, Ringland, Radford, Murphy, & Curran, 2017). Después de esto, se realizó una corrección ortográfica de los *tokens* con el paquete *Spell-Checker* (Pyspellchecker, 2021),

[4] es_core_news_lg

limpieza de las oraciones (remover espacios dobles, signos de puntuación u otros caracteres), normalización del texto (convertir palabras a minúsculas y eliminar tildes) y finalmente la eliminación de palabras vacías incluidas dentro de la lista marcada por *Spacy* u otras palabras consideradas como *stop words* como "a" e "y" también fueron eliminadas.

IV. MODELAMIENTO

Clasificador de oraciones: De acuerdo con las recomendaciones de la literatura (Terechshenko, 2021) se desarrollaron 3 grupos de modelos según las aproximaciones existentes para la clasificación de textos: modelos basados en conteos de palabras (*bag of words - BOW*), vectores de palabras (w*ord embeddings*) y modelos de lenguaje (*transfomers*). Como métricas de desempeño se usaron los indicadores mencionados en la *Tabla 1,* donde los umbrales mínimos aceptados deberán ser superiores al 90% (para el área bajo la curva) y superiores al 80% para las demás métricas. En total se entrenaron 23 modelos divididos en cuatro grupos según su alcance, seleccionando de manera aleatoria[5] el 70% de las oraciones para entrenamiento y el 30% restante para prueba. A continuación, se muestra la cantidad de oraciones que quedó asignada en cada subconjunto para cada una de las categorías.

Tabla 6. Partición entrenamiento (70%) y prueba (30%)

		Train	***Test***
Tamaño (n)	**Causa (0)**	3221	1373
	Problema (1)	1125	458
	Efecto (2)	3663	1545
	***Dummy* (3)**	1969	901

[5] Utilizando el número 1995 como semilla arbitraria para asegurar la reproducibilidad en la partición del conjunto de datos.

4.1 Modelos Ingenuos basados en *Bag of words* (BOW)

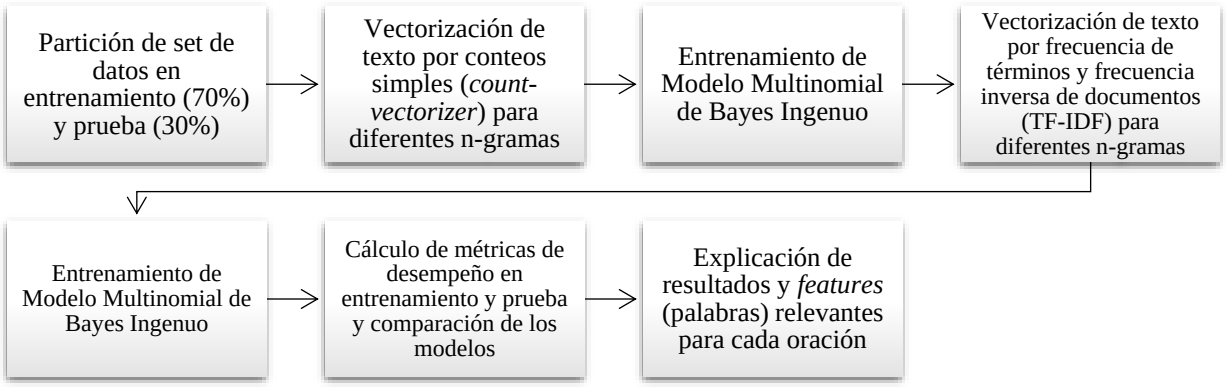


Ilustración 10. Pipeline para los modelos ingenuos basados en bag of words (BOW)

Se utilizó un modelo de Bayes Ingenuo Multinomial utilizando diferentes tipos de vectorización de palabras bajo el paradigma de BOW. Este paradigma recibe dicho nombre debido a que ignora la composición semántica y sintáctica de las oraciones, por lo que cada palabra hace parte de un conjunto de características aisladas o bolsa de palabras. Se escogió este modelo ingenuo ya que computacionalmente es muy eficiente para manejar matrices dispersas de alta dimensionalidad, agilizando el proceso de experimentación para diferentes combinaciones de n-gramas. Las vectorizaciones escogidas estuvieron basadas en conteos simples (*count vectorizer*) y conteos normalizados por la frecuencia de los términos y la frecuencia inversa de documentos (TF-IDF). Para ambos tipos de vectorización, se utilizaron todas las combinaciones posibles de n-gramas en el rango (1,3). Dado que en la literatura no se tiene un referente que sirva como punto de comparación para la clasificación de oraciones en árboles de problemas, este tipo de vectorizaciones basadas en conteos serán el primer punto de partida para el ejercicio de clasificación y permitirán cuantificar el valor agregado de modelos más complejos dentro del *pipeline*.

A continuación, se muestran dos grupos de tablas, el primero muestra los parámetros utilizados para realizar la vectorización según el caso (*count-vectorizer* o TF-IDF*)* y los parámetros del modelo multinomial de Bayes Ingenuo. Para la vectorización y el modelo ingenuo, estos parámetros hacen referencia a la documentación de la librería *scikit-learn* (Scikit-learn, 2021). Por último, se muestra un resumen de las métricas de desempeño para los diferentes n-gramas probados:

Tabla 7. Parámetros utilizados para la vectorización y el modelo ingenuo, respectivamente

Parámetros	Vectorizador	
	Count Vectorizer	*TF-IDF*
Máx document frequency	1.00	1.00
Mín. document frequency	1.00	1.00

Parámetro	Model: Multinomial Naïve Bayes
Smoothing (α)	1.00

Tabla 8. Métricas de desempeño modelos ingenuos BOW

Vectorización (n-grama)													
Model: Multinomial Naïve Bayes		**Count Vectorizer (1,1)**		**Count Vectorizer (1,2)**		**Count Vectorizer (2,2)**		**Count Vectorizer (1,3)**		**Count Vectorizer (2,3)**		**Count Vectorizer (3,3)**	
		Train	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*
Métrica	Accuracy	0.88	0.84	0.93	0.86	0.93	0.81	0.94	0.87	0.94	0.81	0.84	0.68
	Precision	0.88	0.84	0.93	0.86	0.93	0.82	0.94	0.87	0.94	0.82	0.88	0.78
	Recall	0.88	0.84	0.93	0.86	0.93	0.81	0.94	0.87	0.94	0.81	0.84	0.68
	F1-score	0.88	0.84	0.93	0.86	0.93	0.81	0.94	0.87	0.94	0.81	0.84	0.67
	AUC	0.97	0.95	0.99	0.96	0.99	0.95	0.99	0.96	0.99	0.95	0.97	0.88

Vectorización (n-grama)													
Model: Multinomial Naïve Bayes		**TF-IDF (1,1)**		**TF-IDF (1,2)**		**TF-IDF (2,2)**		**TF-IDF (1,3)**		**TF-IDF (2,3)**		**TF-IDF (3,3)**	
		Train	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*
Métrica	Accuracy	0.88	0.84	0.91	0.85	0.90	0.79	0.92	0.85	0.90	0.78	0.78	0.64
	Precision	0.88	0.85	0.92	0.86	0.91	0.81	0.92	0.86	0.91	0.81	0.84	0.76
	Recall	0.88	0.84	0.91	0.85	0.90	0.79	0.92	0.85	0.90	0.78	0.78	0.64
	F1-score	0.88	0.84	0.91	0.85	0.90	0.78	0.92	0.85	0.90	0.78	0.77	0.62
	AUC	0.97	0.96	0.98	0.96	0.99	0.94	0.98	0.96	0.99	0.94	0.97	0.88

De todas las aproximaciones posibles, la aproximación más básica sería usar una vectorización *count-vectorizer* con uni-gramas de palabras (1,1) tal como se muestra resaltado en azul oscuro. Por este motivo y sumado el hecho de que no existe dentro de la literatura un punto de comparación relevante, se utilizara de ahora en adelante esta vectorización con Bayes Ingenuo como *benchmark* para cuantificar el valor agregado de otros modelos.

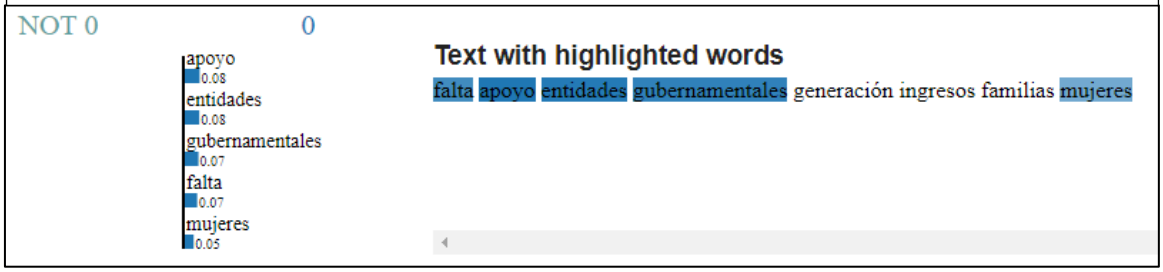
Dado el contexto de los árboles de problemas, tiene bastante sentido trabajar con bi-gramas o incluso tri-gramas, pues por construcción, muchas de las oraciones están redactadas con negaciones del estilo: “ausencia de” o “falta de”, así como el uso de varias entidades nombradas como colegios o municipios. Esta intuición se ve reflejada en métricas mejores para las vectorizaciones que incorporan bi-gramas (1,2) o tri-gramas (1,3) frente a únicamente uni-gramas (1,1). Los mejores resultados para cada grupo de vectorización aparecen resaltados en rojo, por lo

que es posible concluir que entre la vectorización por conteos simples o TF-IDF, se prefiere la primera con utilización de n-gramas (1,3), pues mejora levemente todas las métricas de desempeño del modelo.

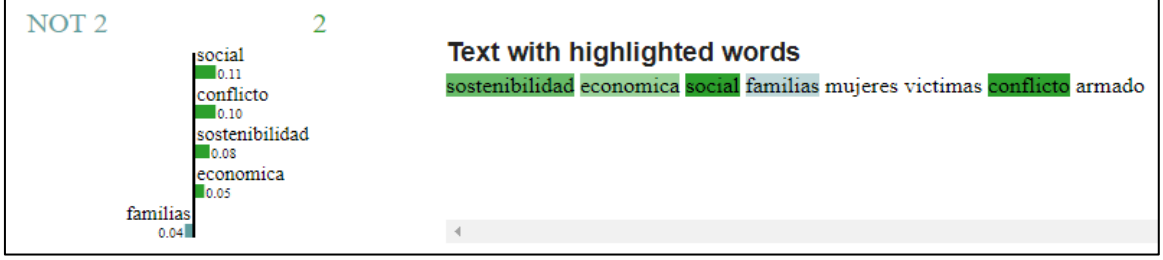
Explicación de resultados: Independientemente de las métricas, una de las ventajas de estos modelos sencillos, es que es posible observar de manera directa aquellas características (palabras) que hacen que el modelo clasifique una oración dentro del árbol de problemas. Esto último utilizando técnicas como LIME[6] (Ribeiro, 2016). Tomando como ejemplo el modelo Bayes Ingenuo *count-vectorizer* (1,3), se pueden entender las reglas (pesos de las palabras) bajo las cuales se clasificaron las oraciones (pre-procesadas) pertenecientes a un mismo árbol de problemas escogido arbitrariamente. A continuación, como ejemplo, se muestran los valores reales y predichos para una causa (0), problema (1) y efecto (2) clasificados de acuerdo con una probabilidad {p} asignada.

Ilustración 11. Ejemplo de extracción de características relevantes para modelos de conteo para un árbol de problemas arbitrario

Causa (0): "falta apoyo entidades gubernamentales generación ingresos familias mujeres"
Real (0) | Predicho (0) {p = 1.0}



Problema (1): "sostenibilidad economica social familias mujeres victimas conflicto armado"
Real (1) | Predicho (2) {p = 0.73}



[6] *Local interpretable model-agnostic explanations.*

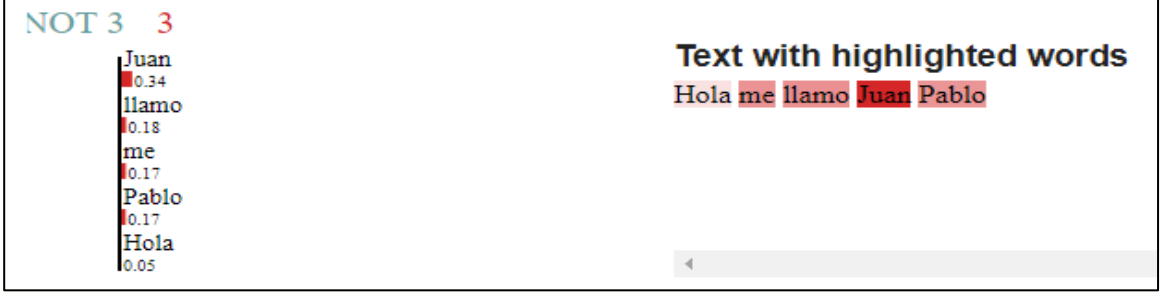
Efecto (2): incremento población condiciones desplazamiento centros poblados
Real (2) | Predicho (2) {p = 0.58}

Se puede observar que para este ejemplo correspondiente a personas víctimas del conflicto armado, que el modelo fue capaz de identificar correctamente esta causa y efecto particular, pero no el problema. Independientemente si el modelo hizo una predicción correcta o incorrecta, lo interesante es ver cuáles fueron las palabras (con sus respectivos pesos) que jalonaron una clase o la otra. Por ejemplo, para el caso de la causa clasificada correctamente, las palabras más relevantes fueron: “apoyo” (0.08), “entidades” (0.08), para el problema clasificado de manera incorrecta como efecto fueron: “social” (0.11), “conflicto” (0.10) y finalmente para el efecto clasificado correctamente fueron: “incremento” (0.42), “desplazamiento” (0.32).

Para el caso de una oración conformada únicamente por ruido, vemos que el modelo es capaz de discernir perfectamente que esta oración pertenece a la clase *dummy* (3):

Ilustración 12. Prueba del modelo con una oración 100% dummy

Dummy (3): Hola me llamo Juan Pablo
Real (3) | Predicho (3) {p = 0.1}



Pensando en el objetivo de negocio, el cual busca coadyuvar el proceso de formulación y validación de árboles de problemas, es de resaltar que esta interpretabilidad y visualización no es menor aun cuando se trata de un modelo “ingenuo”. Pues en el ejercicio de validación se puede

preferir un modelo interpretable con indicadores de desempeño aceptables sobre un modelo de "caja negra" con indicadores de desempeño más altos.

4.2 *Word embeddings*

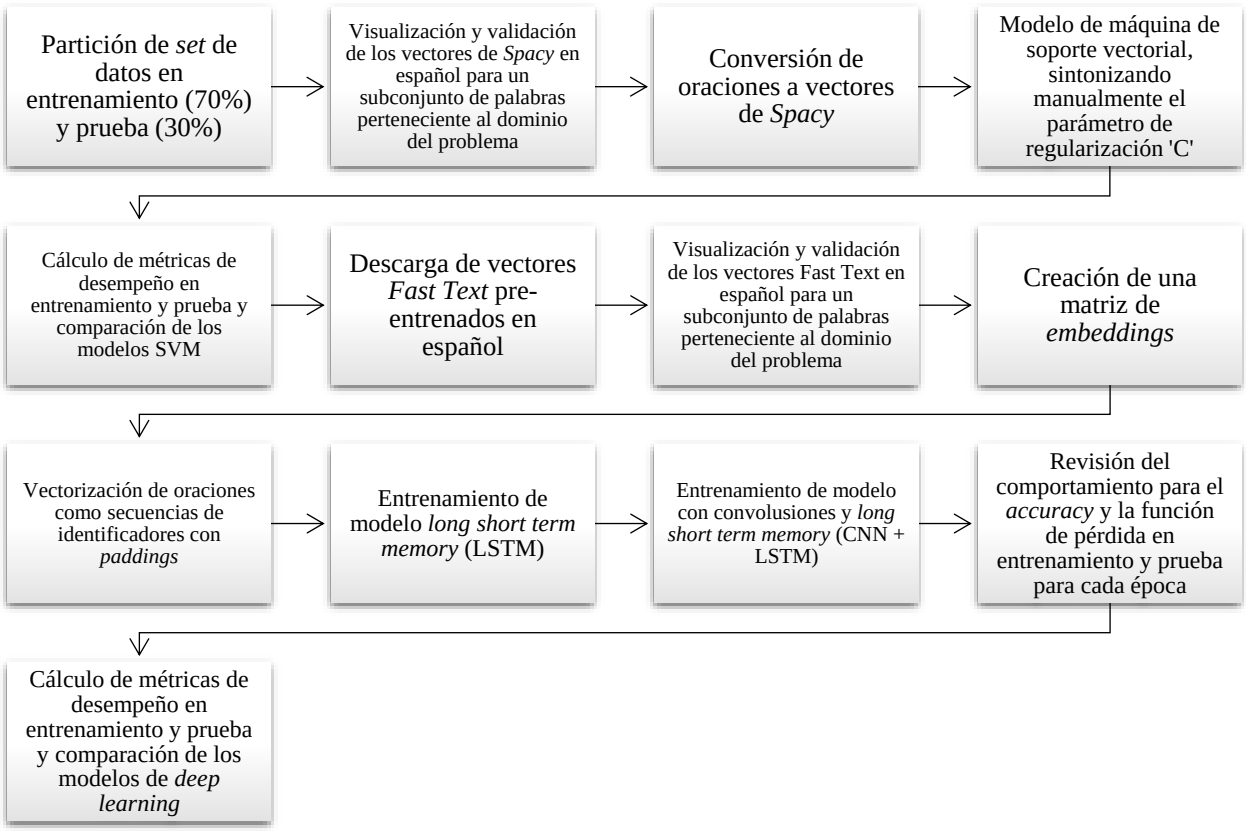


Ilustración 13. Pipeline para los modelos basados en Word Embeddings

Se utilizaron dos aproximaciones para el uso de vectores de palabras: **1)** modelos basados en máquinas de soporte vectorial utilizando los vectores precargados[7] que existen por defecto dentro del *pipeline* de PLN de *Spacy* y **2)** modelos de aprendizaje profundo basados en *Transfer Learning*. En ambos casos, el paradigma bajo el cual los algoritmos incorporan las palabras es fundamentalmente distinto a los modelos BOW, pues a partir de representaciones vectoriales de palabras, los modelos son capaces de incorporar la información semántica de las mismas.

Máquinas de soporte vectorial (SVM): Es uno de los clasificadores más populares para la clasificación de textos, debido a que: **1)** es capaz de modelar comportamientos no lineales y es más robusto al sobre ajuste cuando hay una alta dimensionalidad en las variables (Kowsari, 2019) y **2)**

[7] Estos vectores precargados tienen un tamaño de 300 dimensiones.

tiene un bajo tiempo de procesamiento y buen rendimiento que supera a otros modelos ingenuos (Mihaela, 2019). Este algoritmo busca encontrar el mejor hiperplano que separe las clases maximizando el margen (distancia) entre ellas. Este tipo de modelo puede manejar tareas de clasificación lineales y no lineales; esto último se puede lograr usando diferentes *kernel*, los cuales pueden ser de tipo radial o polinomial.

Este algoritmo incluye un hiper parámetro conocido como 'C', el cual controla el número y la severidad de violación al margen e hiperplano que se toleran durante el proceso de ajuste. Cuando 'C' tiene un valor muy grande no permite ninguna violación a esta regla, y en el caso contrario cuando es bajo, no penaliza severamente a los errores. Siendo así se puede decir que 'C' es un parámetro de regularización que se encarga de encontrar el balance entre sesgo y varianza del modelo (Amat, 2017).

Tal como se mencionó en la sección 3, el corpus utilizado dentro del *pipeline* de *Spacy* proviene del *AnCora* (Taulé, 2008) y *WikiNER* (Nothman, Ringland, Radford, Murphy, & Curran, 2017). Al utilizar técnicas de reducción de dimensionalidad, es posible visualizar un subconjunto arbitrario de palabras ("vereda", "municipio", "carretera") perteneciente al contexto que se está trabajando, y de esta manera, entender de forma más intuitiva las relaciones que existen entre las palabras de este recurso (anexo 2). Tomando como ejemplo la palabra "vereda", es posible observar que esta está cercana a las palabras "carreterita", "veredal", "loma" o "huacalera" y así sucesivamente las otras asociaciones para las demás palabras.

Para poder incorporar estos vectores a los modelos de SVM, se realizó la vectorización de las oraciones utilizando el vector promedio del documento, calculado a partir de los vectores individuales de cada palabra. Esta aproximación tan sencilla permite reducir bastante la dimensionalidad de la matriz de características, lo cual es una gran ventaja frente a las matrices dispersas del paradigma BOW, por lo que la matriz resultante será un arreglo de 300 columnas y 9.978 filas, correspondientes al tamaño de los vectores y el número de documentos de entrenamiento, respectivamente.

En esta fase de experimentación se mantuvo fijo el tipo *kernel* utilizado, variando el hiper parámetro de regularización 'C'. En este caso se utilizó un *kernel* radial con el fin de capturar fronteras de decisión no lineales, utilizando el valor de gamma (γ) definido por defecto por la librería de s*cikit-learn*, el cual esta escalado dependiendo de las características del conjunto de

datos de entrenamiento (Scikit learn, 2021) y se varió ‘C’ en el intervalo discreto de [0.1, 1, 10, 100, 1000], tal como lo resume la siguiente tabla de parámetros y tabla de resultados:

Tabla 9. Parámetros máquinas de soporte vectorial

	Hiper-parámetro de regularización 'C'				
Parámetros	**C = 0.1**	**C = 1**	**C = 10**	**C = 100**	**C = 1000**
Kernel	radial	radial	radial	radial	radial
Gamma	0.00244	0.00244	0.00244	0.00244	0.00244

Tabla 10. Resultados de los modelos SVM usando los vectores de Spacy

Model: Support Vector Classifier		**Hiper-parámetro de regularización 'C'**									
		C = 0.1		**C = 1**		**C = 10**		**C = 100**		**C = 1000**	
		Train	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*
Métrica	Accuracy	0.83	0.83	0.89	0.87	0.96	0.89	0.99	0.88	0.99	0.88
	Precision	0.83	0.83	0.90	0.87	0.96	0.89	0.99	0.88	0.99	0.88
	Recall	0.83	0.83	0.89	0.87	0.96	0.89	0.99	0.88	0.99	0.88
	F1-score	0.83	0.82	0.89	0.87	0.96	0.89	0.99	0.88	0.99	0.88
	AUC	0.96	0.96	0.98	0.97	0.99	0.97	1.00	0.97	1.00	0.97

Este parámetro de regularización ‘C’ es muy importante para el desempeño de los modelos pues de acuerdo con lo mencionado anteriormente “intercambia la clasificación correcta de los ejemplos de entrenamiento con la maximización del margen de la función de decisión” (Pedregosa, 2011).

En un escenario ideal, sería deseable hacer una búsqueda de todos los hiper parámetros de forma simultánea (*kernel*, *gamma*, costo) por medio de un *grid search*, por ejemplo. No obstante, hacer una búsqueda exhaustiva de este tipo es computacionalmente costoso, especialmente para los modelos de máquina de soporte. Teniendo esto en cuenta y dado que las aproximaciones planteadas hasta el momento no presentan diferencias muy significativas, se tomará esta alternativa de probar valores extremos a modo de ensayo y error. Del resultado anterior se concluye que un valor de C=10 parece ser el punto medio de regularización adecuado con mejores métricas de desempeño en prueba.

Transfer Learning: Se utilizaron recursos propios del idioma español, específicamente los *embeddings* (Cañete, 2021) creados a partir de un modelo *FastText* (Bojanowski, 2017) entrenado con el *Spanish Unannotated Corpora* (SUC). Este corpus posee un total de casi 3 billones de palabras pertenecientes a 15 recursos diferentes en español (*Wikipedia, ParaCrawl, GlobalVoices*,

entre otros) para un total de 1.3 millones de vectores con tamaño de 300. Una de las ventajas de estos vectores fue que fueron obtenidos a partir del algoritmo *FastText*, lo cual es ideal para este contexto de negocio en particular, frente a otras aproximaciones como *Word2Vec* (Mikolov, 2013) y *GloVe* (Pennington, 2014) por las siguientes razones:

1) Los vectores obtenidos son más robustos cuando el *set* de datos de entrenamiento para la clasificación es limitado. Esto es particularmente cierto para este caso, pues la base de datos de árboles de problemas es limitada.

2) Al utilizar n-gramas de caracteres, el modelo es capaz de asignar representaciones vectoriales a palabras OOV (*out-of-vocabulary*), es decir, palabras por fuera de la base de datos de entrenamiento con la cual se crearon los vectores. Esto resulta bastante útil, pues este contexto particular de árboles de problemas está bastante acotado a un universo muy particular de proyectos en Cundinamarca. Adicionalmente, hay muchas palabras que están mal escritas, por lo que este algoritmo permite darles representación a estas palabras aun cuando estén mal escritas.

3) Al incorporar la sub-información de las palabras, los vectores obtenidos son mejores, particularmente para idiomas morfológicamente ricos, por ejemplo: alemán, turco, árabe y ruso y en alguna medida español.

A pesar de que este tipo de *embeddings* sean idóneos para este contexto, también es importante tener en cuenta que una de sus principales limitaciones es que al utilizar n-gramas de caracteres, estos vectores no tienen en cuenta la posición de las palabras y por ende se pierde su significado sintáctico dentro de la vectorización. No obstante, por todos los beneficios mencionados anteriormente se decidió utilizarlos.

Al visualizar el mismo subconjunto arbitrario de palabras (“vereda”, “municipio”, “carretera”) que se mostró para los vectores de *Spacy*, es posible entender de manera más intuitiva las relaciones que existen entre las palabras de este recurso (anexo 3). A diferencia de los vectores de *Spacy*, aquí se hace mucho más evidente la distancia existente entre la palabra “municipio” y las palabras “vereda”, “carretera”, dando a entender que estas dos últimas son más parecidas entre ellas. Nótese como en este caso como la palabra municipio tiene una representación muy parecida al recurso anterior con palabras como: “municipiu”, “delmunicipio”, “municipia”, “elmunicipio”. En el caso de la palabra “vereda”, la relación sería con: “plazuela”, “veredón”, “veredad” y

finalmente para “carretera”: “carrretera”, “lacarretera”, “autovía”, “carreteras”, etc. En síntesis, para ambos recursos se tienen relaciones que hacen sentido para el idioma español y para el contexto de negocio que se está trabajando, por lo que se decidió incorporar la información de estos vectores dentro del ejercicio de clasificación.

Para poder utilizar estos vectores pre-entrenados dentro de los modelos de redes neuronales profundas que se explicarán posteriormente, es necesario realizar dos pasos: **1)** crear una matriz de *embeddings* que servirá como insumo para crear la capa de entrada de la red neuronal y **2)** asignar un identificador (*id*) único a cada palabra, de tal modo que cada oración tenga una representación numérica de acuerdo con estos *id*.

La matriz de *embeddings* es un arreglo de dos dimensiones que contiene los vectores asociados a las palabras de los documentos de entrenamiento. El número de columnas es igual al tamaño de los vectores (300) y las filas corresponden al número de palabras únicas (vocabulario) en el corpus de entrenamiento. La capa de entrada de la red neuronal es un tipo de *embedding layer* de *Keras* (Keras, 2021) que convierte las oraciones a vectores densos. En este caso, los pesos de de la capa corresponden a la matriz de embeddings (*weights*), la dimensión de entrada corresponde al número de filas (*input_dim*) y la dimensión de salida (*output_dim*) al número de columnas de la matriz. Por último, es necesario especificarle a esta capa de entrada la longitud de entrada de las oraciones (*input_length*). Para que todas las oraciones tuvieran la misma longitud se realizó un proceso de *padding* con ceros hasta que cada oración alcanzará una longitud máxima igual al tamaño del documento más largo de entrenamiento (81).

Una vez explicado el racional de estos vectores, es necesario revisar de acuerdo con la literatura existente, las posibles aproximaciones desde la óptica del *Deep Learning* para el uso de diferentes tipos de redes neuronales profundas. Una posible aproximación consiste en utilizar algoritmos de aprendizaje profundo para apalancar el uso de recursos externos dentro del corpus propio por medio de *Transfer Learning*. De acuerdo con (Yin, 2017) esto se puede hacer por medio de redes neuronales recurrentes del tipo l*ong-short term memory* (LSTM) o *gated recurrent unit* (GRU) o redes *feed-forward* como las redes convolucionales. En el campo del procesamiento de lenguaje natural aplicado a la clasificación de textos, ambos tipos de redes presentan ciertos tipos de ventajas y desventajas que hacen que, dependiendo del contexto, valga la pena utilizar una o la

otra, o inclusive, una combinación de ambas. A continuación, se muestra una breve explicación de las posibles ventajas/desventajas de acuerdo con lo que mencionan estos autores.

En el caso del PLN hace sentido utilizar redes neuronales recurrentes, pues el lenguaje natural es secuencial y tiene un contexto. Por definición, este tipo de redes son capaces de capturar comportamientos temporales de manera secuencial, lo que las hace ideales para trabajar con tareas de PLN. No obstante, trabajar de manera secuencial no siempre es apropiado pues el contexto y orden de las palabras puede llegar a inducir sesgos, por lo que otra alternativa son las redes neuronales convolucionales. Dentro de los casos de uso más convencionales, este tipo de redes son usadas para la clasificación de imágenes, ya que estas son capaces de detectar características locales relevantes e invariantes en su posición. En el caso del PLN, estas características serían conjuntos de palabras clave en formas de n-gramas que no dependen de la posición dentro del documento, ya que el orden local de las palabras se pierde al aplicar convoluciones y operaciones de *pooling*. No obstante, al realizar esta reducción de dimensionalidad, se logran abstraer las características más relevantes y reducir los tiempos de entrenamiento.

En síntesis, la idoneidad de cada red dependerá del objetivo de la minería de datos y el corpus utilizado. Para este caso particular de árboles de problemas hace más sentido trabajar con redes neuronales recurrentes por dos razones principales: **1)** el *set* de datos es corto y no hay tiempos de entrenamiento largos y **2)** el contexto de las oraciones y su longitud es relevante. No obstante, dentro de esta fase de experimentación se probará una red LSTM y una mezcla de una red LSTM con una red convolucional. A continuación, se muestra la topología de cada red, con sus respectivos parámetros y resultados:

Tabla 11. Topología de las redes neuronales de Word Embeddings

LSTM	**CNN + LSTM**
Layer (type) Output Shape Param # embedding_2 (Embedding) (None, 81, 300) 2167800 dropout_1 (Dropout) (None, 81, 300) 0 lstm_1 (LSTM) (None, 32) 42624 dense (Dense) (None, 4) 132 Total params: 2,210,556 Trainable params: 42,756 Non-trainable params: 2,167,800	Layer (type) Output Shape Param # embedding_2 (Embedding) (None, 81, 300) 2167800 conv1d (Conv1D) (None, 81, 300) 180300 max_pooling1d (MaxPooling1D) (None, 40, 300) 0 dropout_2 (Dropout) (None, 40, 300) 0 lstm_2 (LSTM) (None, 32) 42624 dense_1 (Dense) (None, 4) 132 Total params: 2,390,856 Trainable params: 223,056 Non-trainable params: 2,167,800

Tabla 12. Parámetros de las redes neuronales

		Modelo	
		LSTM	**CNN + LSTM**
Parámetro	**Número de neuronas LSTM**	32	32
	Learning rate	0.01	0.01
	Épocas	25	10
	Batch size	32	32
	Función de activación LSTM	tanh	tanh
	Función de activación Conv1D	N/A	ReLu
	Función de activación capa de salida	softmax	softmax
	Filtros	N/A	300
	Kernel	N/A	2
	Dropout	0.2	0.2
	Dropout LSTM	0.2	0.5
	Recurrent dropout LSTM	0.2	0.5
	Métrica	Accuracy	Accuracy
	Función de pérdida	Sparse categorical crossentropy	Sparse categorical crossentropy
	Optimizador	Adam	Adam

Con respecto a los modelos anteriores, se decidió ser muy cauteloso en la definición de los parámetros de entrenamiento, pues dentro de las primeras iteraciones de esta fase de experimentación se descubrió que estas redes tienden a sobre ajustarse rápidamente, por lo que resultaba importante no excederse en la cantidad neuronas de la capa LSTM, ni en las épocas de entrenamiento, así como realizar algún tipo de regularización. Siendo así, se decidió agregar una capa especifica de *dropout* así como un *dropout* y *dropout* recurrente en la capa LSTM de ambos modelos.

Dado que conceptualmente la red CNN + LSTM realiza algo similar a una reducción de dimensionalidad al aplicar convoluciones, esta red tiende a sobre ajustarse más rápidamente que la anterior, por lo que en este caso se decidió aumentar los parámetros de regularización, así como entrenarla por una menor cantidad de épocas. El *kernel* igual a 2 se escogió con el fin de capturar bi-gramas, pues en los modelos BOW se demostró que estas diadas de palabras aportan al entendimiento de las oraciones para el modelo. El número de filtros es igual al tamaño de los vectores (300). El optimizador *adam* fue escogido debido a su popularidad para este tipo de redes, mientras que la función de pérdida fue escogida dado que se trata de un problema de clasificación multinomial. Una vez definidos los parámetros anteriores, se entrenaron los modelos por 25 (LSTM) y 10 épocas (CNN + LSTM), logrando así tener un balance razonable entre sesgo-varianza

tal como lo refleja el historial de entrenamiento de ambas redes y las métricas de desempeño para entrenamiento y pruebas presentadas a continuación.

Ilustración 14. Comportamiento por épocas de la red LSTM en entrenamiento y prueba

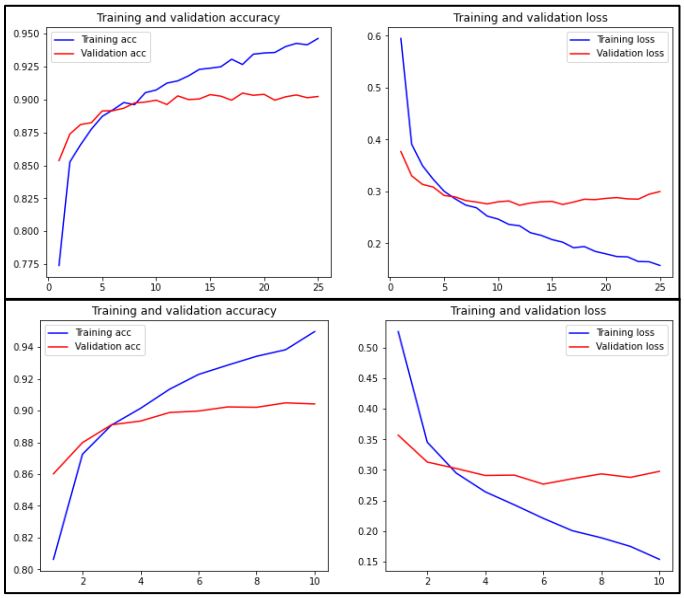


Ilustración 15. Comportamiento por épocas de la red CNN + LSTM en entrenamiento y prueba

Tabla 13. Métricas de desempeño de las redes neuronales

		Modelo			
		LSTM		CNN + LSTM	
		Train	*Test*	*Train*	*Test*
Métrica	**Accuracy**	0.96	0.90	0.97	0.90
	Precision	0.96	0.90	0.97	0.90
	Recall	0.96	0.90	0.97	0.90
	F1-score	0.96	0.90	0.97	0.90
	AUC	1.00	0.98	1.00	0.98

Al observar los resultados de la tabla anterior se ve que las métricas de desempeño en prueba son exactamente iguales para ambos modelos, con la salvedad de que la red CNN + LSTM tiene una mayor brecha entre las métricas de entrenamiento y prueba, lo cual muestra que está más sobre ajustada frente a la red LSTM. Para este caso particular parece ser que aplicar convoluciones no genera ningún valor agregado en términos de mejorar las métricas de desempeño, pero si se sobre ajusta rápidamente a los datos, generando una mayor brecha entre los resultados de entrenamiento y prueba, aún con una mayor regularización y menos épocas. Una posible hipótesis al respecto es que al tener oraciones que, en general, son más bien cortas, esto no genera ningún tipo de abstracción de características relevantes para el modelo, lo cual resulta en un mayor sobre ajuste. Siendo así y utilizando el principio de la Navaja de Ockham, se prefiere el modelo más sencillo, pues en igualdad de condiciones tiene menor varianza frente al modelo más complejo.

4.3 Modelo de lenguaje (*transformer*)

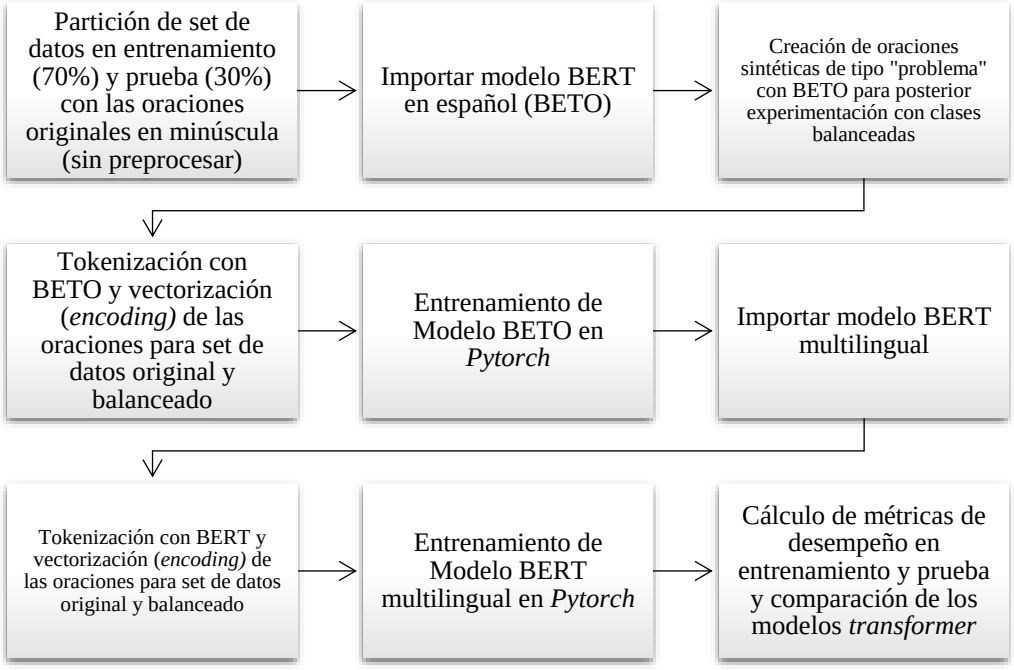


Ilustración 16. Pipeline modelos transformer

Para los modelos de lenguaje se utilizaron aproximaciones basadas en *transfer learning* de *Bidirectional Encoder Representations from Transformers* (BERT) (Devlin, Pre-training of deep bidirectional transformers for language understanding, 2018). Se utilizaron dos recursos provenientes del repositorio de *Hugging Face*, el primero fue un modelo BERT multilingüe pre-entrenado[8] (Devlin, Bert: Pre-training of deep bidirectional transformers for language understanding., 2018) y el segundo un modelo BERT en español - BETO[9] (Canete, 2020).

Tal como su nombre lo indica, estos modelos son bidireccionales, lo cual significa que tienen la capacidad de analizar una oración en dos direcciones para cada palabra y de esta manera entender en profundidad el contexto y la temática de cada frase, de manera muy similar a como lo haría un ser humano. A diferencia de las aproximaciones anteriores basadas en vectores de palabras, este entendimiento del lenguaje es bastante valioso por ejemplo para palabras polisémicas, pues su significado subyacente depende del contexto de la frase. Para este caso particular, dichas palabras tendrán una representación vectorial igual aun cuando el contexto sea

[8] https://huggingface.co/bert-base-multilingual-uncased
[9] https://huggingface.co/dccuchile/bert-base-spanish-wwm-uncased

diferente, mientras que el modelo *transformer* si es capaz de capturar esta heterogeneidad en el significado dependiendo del contexto. Otra ventaja importante de estos modelos es que, en general, son capaces de superar a la mayoría de los modelos actuales en su capacidad predictiva aún con poco o ningún preprocesamiento de las oraciones.

Teniendo todo lo anterior en cuenta, estos dos modelos base fueron sobre ajustados en *pytorch* con la ayuda de una GPU, utilizando la documentación sugerida proveniente de *Transformers* (Hugging Face, 2021) de tal manera que estos estuvieran afinados a la tarea específica de clasificación de árboles de problemas. Adicionalmente y de acuerdo con la literatura (Wei, 2019), otras de las ventajas de estos modelos *transformer*, es que es posible apalancarse de la generación de oraciones sintéticas de estos modelos pre-entrenados para mejorar las métricas de clasificación de clases desbalanceadas. Como se mencionó anteriormente la construcción de árboles de problemas es por defecto desbalanceada en lo que respecta al problema central con relación a sus causas-efectos, por lo que como parte del proceso de experimentación se decidió realizar un proceso de *over-sampling* de esta clase minoritaria con el fin de corroborar si las métricas de esta clase en particular mejoran o no. Teniendo en cuenta el *pipeline* de (Wei, 2019) se utilizó un parámetro $\alpha = 0.1$, para generar nuevas oraciones sintéticas con la ayuda del modelo BETO, reemplazando aproximadamente el 10% de las palabras en las oraciones existentes de la base de datos de entrenamiento. A continuación, se muestran algunos ejemplos de las oraciones generadas por esta vía después de convertir a minúscula todas las palabras:

Tabla 14. Oraciones sintéticas generadas a partir del transformer BETO

Oración "Problema" Original	**Oración "Problema" Sintética**
ESCASAS INICIATIVAS PRODUCTIVAS PARA INCENTIVAR A FAMILIAS VICTMAS DEL CONFLICTO PARA LA GENERACION DE INGRESOS	promuevan iniciativas productivas para incentivar a familias victmas del conflicto para la generacion de ingresos
Deficiencia en las reglas, protocolos y procesos de la corresponsabilidad entre la Nación y demás entes territoriales del país en la implementación de la política para víctimas y demás actores del conflicto	deficiencia en las reglas, protocolos y procesos de la corresponsabilidad entre la nación y nuevos colectivos territoriales obso país en la implementación de la política para víctimas y demás actores del conflicto
BAJA DOTACIÓN DE MAQUINARIA, EQUIPOS AGROPECUARIOS Y AGROINDUSTRIALES PARA LA MODERNIZACIÓN DEL SECTOR AGROPECUARIO EN EL DEPARTAMENTO DE CUNDINAMARCA	baja dotación ii maquinaria, equipos agropecuarios y agroindustriales para la modernización del sector agropecuario en el departamento cordillera cundinamarca
BAJA CALIDAD DE LAS CARNES Y SUBPRODUCTOS DEBIDO PROCESO DE ALMACENAMIENTO DE EN CANAL.	mala calidad de las carnes y subproductos debido proceso de almacenamiento de en canal.
Bajos ingresos para los pescadores artesanales, provenientes de la pesca, su actividad principal	bajos impuestos para los pescadores artesanales, provenientes de la pesca, su actividad principal

Originalmente los modelos estaban usando 1125 oraciones de “problema” para entrenamiento y 458 para prueba, no obstante, con la ayuda del modelo *transformer* se logró triplicar la cantidad de oraciones “problema” de tal manera que esta categoría estuviese balanceada con respecto a las oraciones de “causa” y “efecto” en el subconjunto de entrenamiento. En el subconjunto de prueba se mantuvo la proporción original de las oraciones con el fin de no sesgar el desempeño del modelo a un escenario que no es real, pues por defecto siempre habrá más oraciones de tipo “causa” y “efecto” en un escenario de validación real. La siguiente tabla resume ambas particiones realizadas para los dos modelos BERT utilizados:

Tabla 15. Partición de datos en entrenamiento y prueba para los escenarios balanceado y desbalanceado

		Set de datos			
		Balanceado		**Original**	
		Train	***Test***	***Train***	***Test***
Tamaño (n)	**Causa (0)**	3221	1373	3221	1373
	Problema (1)	3375	458	1125	458
	Efecto (2)	3663	1545	3663	1545
	***Dummy* (3)**	1969	901	1969	901

Nótese que en este caso nunca se realizó balanceo para la clase 3 pues está es una categoría *dummy* cuyo propósito es absorber el ruido no perteneciente a árboles de problemas y dado que, en general, los modelos identifican con facilidad este tipo de oraciones, se decidió no realizar ningún tipo de balanceo sobre esta clase.

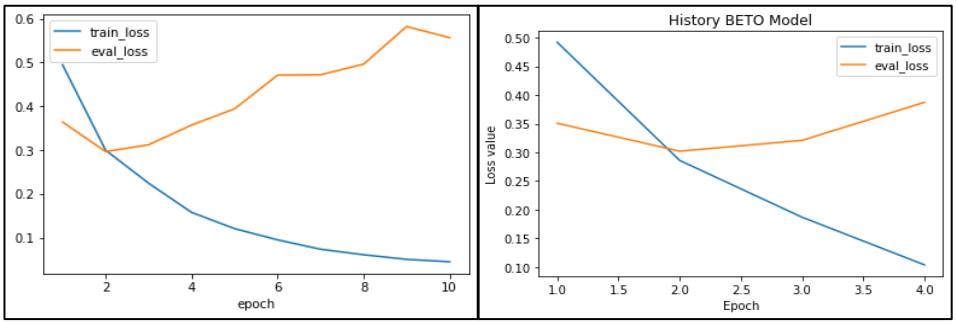
A continuación, se muestran los parámetros utilizados para ambos escenarios de experimentación con diferentes modelos BERT y el balanceo de la clase “problema”.

Tabla 16. Parámetros modelos Transformer

		Modelo			
		BERT Multilingual (Balanceado)	**BERT Multilingual**	**BETO (Balanceado)**	**BETO**
Parámetro	**Warmup steps**	500	500	500	500
	Weight decay	0.01	0.01	0.01	0.01
	Épocas	4	4	4	4
	Learning rate	0.00005	0.00005	0.00005	0.00005
	Optimizer	AdamW	AdamW	AdamW	AdamW
	Max Length	50	50	50	50
	Model	bert-base-multilingual-uncased	bert-base-multilingual-uncased	dccuchile/bert-base-spanish-wwm-uncased	dccuchile/bert-base-spanish-wwm-uncased
	No. Labels	4	4	4	4

En un principio se pensó en entrenar los modelos entre 10 a 20 épocas de acuerdo con las recomendaciones de la literatura para modelos sobre ajustados (Wolf, 2020). No obstante, dentro del proceso de experimentación inicial, fue posible observar que esta aproximación resulta en que el modelo se sobre ajuste rápidamente a los datos. Esto no es deseable e inclusive el historial de entrenamiento soporta está afirmación, pues muestra que la función de pérdida llega inclusive a aumentar conforme el número de épocas aumenta, por lo que se decidió recortar el número de épocas a 4 con el fin de lograr un sano balance entre sesgo y varianza. El comportamiento fue el mismo para los otros modelos (anexo 4).

Ilustración 17. Historial de entrenamiento del modelo BETO para 10 épocas vs. 4 épocas



Una vez recortado el número de épocas, las métricas de desempeño en entrenamiento y prueba fueron las siguientes:

Tabla 17. Métricas de desempeño modelos Transformer

		Modelo							
		BERT Multilingual (Balanceado)		**BERT Multilingual**		**BETO (Balanceado)**		**BETO**	
		Train	***Test***	***Train***	***Test***	***Train***	***Test***	***Train***	***Test***
Métrica	**Accuracy**	0.98	0.91	0.98	0.91	0.99	0.91	0.98	0.91
	Precision	0.98	0.91	0.98	0.91	0.99	0.91	0.98	0.91
	Recall	0.98	0.91	0.98	0.91	0.99	0.91	0.98	0.91
	F1-score	0.98	0.91	0.98	0.91	0.99	0.91	0.98	0.91
	AUC	1.00	0.98	1.00	0.98	1.00	0.98	1.00	0.98

Tal como se puede observar en la tabla anterior, no existen mayores diferencias entre el uso de balanceo o no para la clase problemas en los indicadores globales de los modelos. Haciendo el cálculo de las métricas de desempeño exclusivamente para la clase de "problemas", vemos que si hay una mejora en los indicadores de desempeño, aunque es mínima:

Tabla 18. Comparación modelos (con y sin balanceo) para las métricas de desempeño de la clase "Problema"

		Modelo							
		BERT Multilingual (Balanceado)		BERT Multilingual		BETO (Balanceado)		BETO	
		Train	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*
Métrcias	Precision	1.00	0.91	0.98	0.87	1.00	0.92	0.99	0.87
	Recall	1.00	0.88	0.98	0.88	1.00	0.86	0.99	0.90
	F1-score	1.00	0.90	0.98	0.88	1.00	0.89	0.99	0.88

De cara a la decisión de escoger un modelo *Transformer* cuando todos los indicadores de desempeño son iguales, se usará la evidencia teórica propuesta por (Canete, 2020), la cual argumenta que el modelo BETO es superior al BERT multilingüe[10] para una gran variedad de tareas del procesamiento del lenguaje natural en español, tales como: clasificación, *POS Tagging*, NER, entre otras (ver anexo 5). En conclusión, la recomendación sería inclinarse por el modelo BETO con balanceo (resaltado en rojo), no solo por la evidencia teórica, sino también porque en general la intuición siempre sugiere decantarse por modelos que estén en el idioma original del problema que se está trabajando.

V. EVALUACIÓN

A continuación, se realizan dos tipos de evaluación: **1)** una evaluación teórico-práctica desde la literatura y ciertas consideraciones desde la experiencia de aplicar estos modelos analíticos a los árboles de problemas y **2)** una evaluación de las métricas de desempeño para la tarea de clasificación.

Evaluación teórico-práctica: Tanto las vectorizaciones como los modelos usados en cada caso, presentan una serie de ventajas y desventajas teóricas-prácticas que vale la pena tener en cuenta en conjunto con las métricas de desempeño. Las siguientes tablas adaptadas de (Kowsari, 2019), resumen algunos de estos puntos de una manera muy concreta:

[10] De acuerdo con la literatura disponible hasta octubre de 2019.

Tabla 19. Cuadro comparativo adaptado (Kowsari, 2019) para vectorizaciones Count-vectorizer, TF-IDF y FastText

Vectorizador	Ventajas	Desventajas
Count- vectorizer TF-IDF	• Fácil de calcular. • Fácil de calcular la similitud entre 2 documentos usándolo. • Métrica básica para extraer la mayor cantidad términos descriptivos en un documento. • Las palabras vacías no afectan los resultados cuando se hace la frecuencia inversa de los documentos (IDF).	• No captura la posición en el texto (significado sintáctico). • No capta el significado en el texto (significado semántico).
FastText	• Funciona para palabras *out of vocabulary* - OOV (palabras por fuera del dominio del corpus de entrenamiento) • Funciona con palabras raras (n-gramas raros de caracteres que son compartido con otras palabras)	• No puede captar el significado polisémico de la palabra dentro del texto. • Tiene un alto consumo de memoria para el almacenamiento. • Es computacionalmente más costoso comparando con GloVe y Word2Vec.

Tabla 20. Cuadro comparativo adaptado (Kowsari, 2019) para modelos de Bayes Ingenuo, SVM y Deep Learning

Modelo	Ventajas	Desventajas
Bayes Ingenuo	• Funciona muy bien con datos de texto. • Fácil de implementar. • Rápido en comparación con otros algoritmos	• Fuerte supuestos acerca de la forma de la distribución de los datos (característcas independientes). • Limitado por la escasez de datos, las características nuevas no son reconocidas.
Máquinas de soporte vectorial (SVM)	• SVM permite modelar fronteras de decisión no lineales. • Se desempeña de manera similar a la regresión logística cuando los datos son linealmente separables • Robusto contra problemas de sobreajuste (especialmente para datos de texto, debido a sus altas dimensiones en el espacio)	• Falta de transparencia en los resultados causado por una gran cantidad de dimensiones (especialmente para datos de texto). • Elegir un forma funcional eficiente para el kernel es difícil (susceptible a problemas de sobreajuste / entrenamiento dependiendo del kernel). • Complejidad en el uso de la memoria.
Deep Learning	• Diseño flexible con las características (reduce la necesidad de *feature engeneering* , la cual es un de las tareas que más consumen tiempo en la práctica del aprendizaje automático) • Arquitectura que se puede adaptar a nuevos problemas. • Puede lidiar con entradas y salidas complejas. • Fácil de volver a entrenar los modelos cuando datos más nuevos se hacen disponibles. • Capacidad de procesamiento en paralelo.	• Requiere una gran cantidad de datos (si sólo se tiene una muestra pequeña de datos de texto, es poco probable que estos modelos superen otros enfoques) • El entrenamiento es computacionalmente costoso. • El problema más importante del aprendizaje profundo es la interpretabilidad del modelo (la mayor parte de el tiempo es una caja negra) • Encontrar una arquitectura y estructura eficiente sigue siendo el principal desafío de esta técnica.

Dado que este artículo fue escrito en el 2019, los autores no realizan un análisis similar de los modelos *transformer* (2018), no obstante, de acuerdo con los autores que inventaron estos modelos (Devlin, Bert: Pre-training of deep bidirectional transformers for language understanding., 2018) y la experiencia resultante de implementar estas arquitecturas para varias tareas del PLN en este proyecto, se pueden evaluar varios puntos teórico-prácticos:

Tabla 21. Ventajas y desventajas de la vectorización con los modelos transformer

Vectorizador	Ventajas	Desventajas
Transformer	• Tokenización y vectorización incluida dentro de la arquitectura del modelo. • Entendimiento del contexto de la oración (bidireccionalidad). • Poco o ningún preprocesamiento del texto.	• Se pierde interpretabilidad de las palabras (caja negra).

Tabla 22. Ventajas y desventajas de los modelos Transformer

Modelo	Ventajas	Desventajas
Transformer	• Variedad de recursos para varias tareas del PLN y en varios idiomas desde el repositorio de Hugging Face. • Facilidad de implementar los modelos en pocas líneas de código. • Superan el estado del arte previo en varias tareas del PLN.	• Computacionalmente es costoso de entrenar, el uso de una GPU es altamente recomendable.

Evaluación de las métricas de desempeño de clasificación: Dado que el problema de clasificación es inherentemente desbalanceado por la proporción problema - causas/efectos, se decidió utilizar el *Micro-Average* de las métricas de la *Tabla 1*, el cual es un promedio de las métricas, ponderado por la cantidad de observaciones en cada una. Esta métrica es altamente usada en modelos de clasificación multinomiales desbalanceados (Ajitesh, 2020). A continuación, se muestra una tabla resumen con el modelo que se usó como punto de referencia a la izquierda - Bayes Ingenuo *count-vectorizer* (1,1) – y los mejores modelos a la derecha, según la etapa de experimentación:

Tabla 23. Métricas de entrenamiento y prueba de los mejores modelos según la etapa de experimentación

		Model									
		Bayes Ingenuo Count Vectorizer (1,1)		**Count Vectorizer (1,3)**		**Support Vector Machine (C = 10) Spacy Vectors**		**LSTM**		**BETO (Balanceado)**	
		Train	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*	*Train*	*Test*
Métrica	**Accuracy**	0.88	0.84	0.94	0.87	0.96	0.89	0.96	0.90	0.99	0.91
	Precision	0.88	0.84	0.94	0.87	0.96	0.89	0.96	0.90	0.99	0.91
	Recall	0.88	0.84	0.94	0.87	0.96	0.89	0.96	0.90	0.99	0.91
	F1-score	0.88	0.84	0.94	0.87	0.96	0.89	0.96	0.90	0.99	0.91
	AUC	0.97	0.95	0.99	0.96	0.99	0.97	1.00	0.98	1.00	0.98

Tal como se puede observar en la tabla anterior, el modelo con mejores métricas de desempeño en prueba es el modelo BETO con balanceo. Frente al modelo más ingenuo de todos, vemos que este si otorga un valor agregado en términos del poder predictivo, aún sin realizar casi ningún tipo de pre-procesamiento en las oraciones[11]. Esta conclusión respalda el bagaje teórico detrás de estos modelos en superar el estado del arte existente para muchas tareas del procesamiento del lenguaje natural.

Dado que actualmente no existe un referente en la literatura de los modelos *transformer* aplicados a la clasificación de árboles de problemas, una posible aproximación será comparar las métricas de clasificación de otros modelos sobre ajustados publicados en el repositorio de *Hugging Face*. Esto sirve como punto de referencia, para entender que tan lejos o cerca se está frente al estado del arte alcanzado por otros modelos que siguen una aproximación similar de sobre-ajustarse a una tarea de clasificación particular. La siguiente tabla resume las métricas en datos de prueba alcanzadas por modelos "similares" en español (Hugging Face, 2021) para diferentes tareas de clasificación[12]:

[11] Tal como se explicó en el *pipeline* de los modelos *transformer*, el único preprocesamiento realizado fue convertir las oraciones a minúsculas.
[12] Valores en blanco representan que la métrica no está disponible.

Tabla 24. Métricas de diferentes modelos sobre ajustados a diferentes tareas de clasificación de textos en español

		Tarea						
		Detección de "Fake news"	**Análisis de sentimientos restaurante**	**Detección de "Hate Speech"**	**Clasificación de "Facts"**	**Detección de CyberBullying**	**Análisis de sentimientos**	**Clasifición oraciones pro-independentistas Catalunya**
Métrica	**Accuracy**	0.77	0.86			0.96	0.72	0.79
	Precision				0.83		0.72	
	Recall						0.72	
	F1-score	0.77		0.73	0.85		0.72	

Tal como se puede observar en la tabla anterior, las métricas varían según el tipo de tarea realizada en un rango de 0.77 a 0.96, por lo que es posible afirmar que los resultados del modelo BETO y los otros transformadores son muy satisfactorios para la tarea de clasificación.

Otro hallazgo importante que vale la pena resaltar es que no existen diferencias significativas entre los modelos más sencillos (BOW) y los modelos más sofisticados (*embeddings* & *transformer*). Una de las posibles hipótesis al respecto es que, al tener un *set* de entrenamiento tan pequeño con proyectos reales y aprobados, los cuales pertenecen al mismo contexto social rural de Cundinamarca, los modelos de aprendizaje automático son capaces de detectar patrones en las oraciones y por ende discernir entre causas, problema, efectos y oraciones ajenas a los árboles de problemas sin mucha dificultad. Si en un futuro se quiere lograr una mejor generalización de estos modelos de aprendizaje automático para el contexto colombiano y validar cualquier proyecto independientemente del sector, es necesario contar con una mayor riqueza de oraciones pertenecientes a diversos contextos o inclusive llegar a considerar otras alternativas como *Low-shot* o *Zero-shot Learning* (Yan, 2018) dada la poca disponibilidad de *datasets* anotados. No obstante, esta prueba de concepto muestra que el proceso de validación a través de técnicas de PLN es factible y capaz de obtener buenos resultados.

VI. RECONOCIMIENTO DE ENTIDADES NOMBRADAS

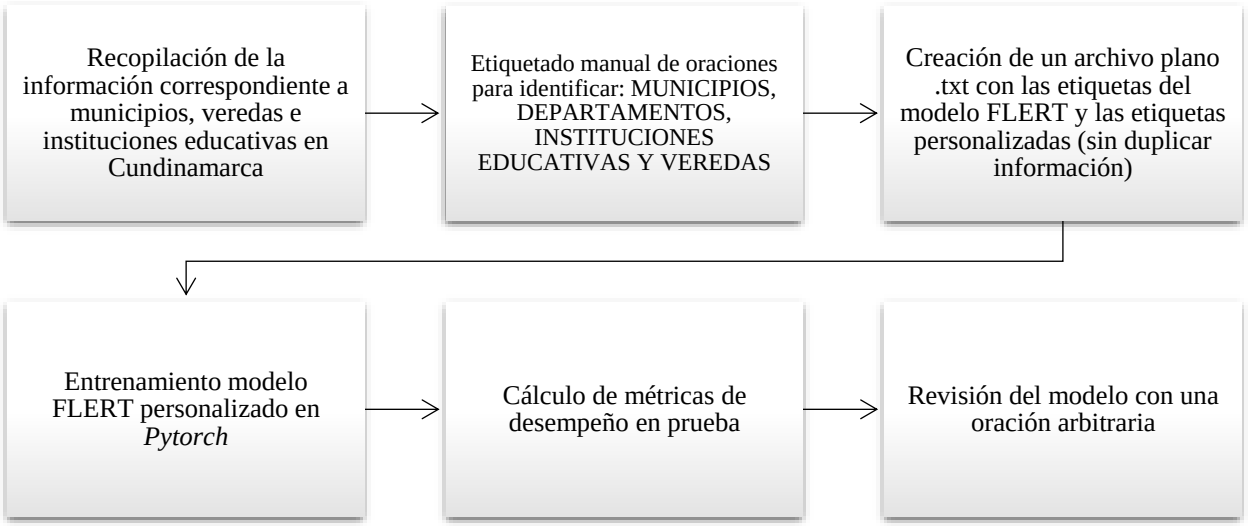


Ilustración 18. Pipeline para la creación de un modelo personalizado de Reconocimiento de entidades nombradas

En la sección de análisis exploratorio se mostró una primera aproximación al reconocimiento de entidades nombradas con el paquete pre-entrenado FLERT (Schweter, Flert: Document-level features for named entity recognition., 2020), no obstante, este modelo tiene sus limitaciones dado el contexto particular que se está trabajando. Por este motivo, a continuación se muestran una serie de pasos a seguir para robustecer este proceso tal como se describe en la documentación de GitHub de estos autores (Schweter, flairNLP / flair, 2021). En este caso sólo se utilizaron las oraciones originales del *set* de datos entregado por IGG y no se incluyó la categoría *dummy*, pues esa categoría sirve dentro del ejercicio de clasificación y no se quiso sesgar este modelo personalizado con oraciones por fuera de los árboles de problemas.

El primer paso consiste en identificar las etiquetas adicionales que se quieran incorporar al modelo de manera personalizada. A partir del análisis exploratorio y dominio del negocio, se sabe que la gran mayoría de entidades nombradas hacen referencia a locaciones en veredas, vías y municipios de Cundinamarca, por lo que el primer paso será subdividir las locaciones bajo las siguientes etiquetas: DEPARTAMENTO, MUNICIPIO y VEREDA. Para lograr esto se hará un etiquetado manual de las dos primeras usando la lista de departamentos y municipios de Colombia disponible en la página de Datos Abiertos (Departamento Administrativo Nacional de Estadística, 2021) mientras que la tercera se hará utilizando los nombres de las veredas registradas en Cundinamarca, disponible en la página de datos abiertos de la gobernación de Cundinamarca

(Infraestructura de Datos Espaciales Cundinamarca IDEC, 2021). Finalmente, otra gran parte de entidades nombradas hacen referencia a instituciones educativas, por lo que el etiquetado manual se hará utilizando los nombres de las instituciones educativas registradas en Cundinamarca ante el Ministerio de Educación Nacional (MEN) y disponibles en el portal "Buscando colegio" (Ministerio de Educación Nacional, 2021). A continuación, se muestran los resultados de cruzar esta información externa con los árboles de problemas del *set* de datos para obtener las etiquetas de DEPARTAMENTO, MUNICIPIO, VEREDA e INSTITUCIÓN EDUCATIVA respectivamente.

Ilustración 19. Etiquetas adicionales para el modelo personalizado FLERT

Al combinar este etiquetado manual con las etiquetas previamente identificadas por FLERT (sin duplicar el etiquetado redundante), se tiene como resultado un conjunto de 6 etiquetas *ad hoc* al contexto de negocio:

Ilustración 20. Nube de palabras para las etiquetas originales y personalizadas

El segundo paso consiste en generar dos archivos planos en formato .txt para realizar las fases de entrenamiento y prueba, donde cada *token* está etiquetado siguiendo el formato (anexo 6) mostrado para los siguientes ejemplos de oraciones: “George Washington went to Washington” y “Sam Houston stayed home” (Chauhan, 2020).

Una vez se tiene la información en el formato requerido por el modelo *transformer*, se procede a entrenar el modelo por 20 épocas utilizando la configuración recomendada en la documentación (Schweter, flairNLP / flair, 2021) con el uso del transformador multilingüe XLM-RoBERTa (Conneau, 2019):

		Modelo **FLERT Personalizado**
Parámetro	**Learning rate**	0.000005
	Mini batch size	4
	Épocas	20
	Scheduler	OneCycleLR
	Weight decay	0
	Model	xlm-roberta-large
	Optimizer	AdamW

Tabla 25. Parámetros modelo personalizado FLERT

A continuación, se muestran las métricas de desempeño en prueba del modelo:

		Etiqueta						
		MUNICIPIO	**VEREDA**	**LOC**	**DEPARTAMENTO**	**MISC**	**ORG**	**INSTITUCIÓN EDUCATIVA**
Métrica	**Precision**	0.84	0.72	0.42	0.93	0.08	0.28	0.37
	Recall	0.88	0.76	0.39	1	0.13	0.25	0.75
	F1-score	0.86	0.74	0.41	0.97	0.1	0.27	0.5

Tabla 26. Métricas de desempeño del modelo personalizado FLERT

Si se toma como ejemplo arbitrario la oración: “los estudiantes del colegio luis gutierrez en la vereda la maría no tienen acceso a internet” se ve que el modelo es capaz de etiquetar correctamente la oración de acuerdo con las etiquetas previamente definidas:

```
los estudiantes del colegio <B-INSTITUCION> luis <I-INSTITUCION> gutier
rez <I-INSTITUCION> en la vereda <B-VEREDA> la <I-VEREDA> maria <I-VERE
                    DA> no tienen acceso a internet
```

Ilustración 21. Ejemplo de detección de entidades nombradas con modelo personalizado para una oración arbitraria

VII. SIMILITUD SEMÁNTICA

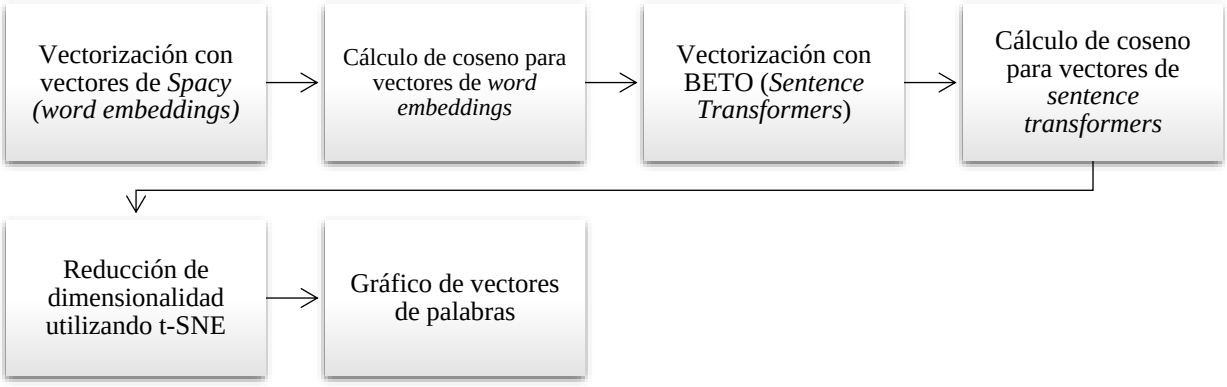


Ilustración 22. Pipeline Similitud Semántica

Para el cálculo de la similitud semántica se decidió apalancarse en dos de las alternativas usadas anteriormente para la clasificación de oraciones: **1)** vectores de palabras usando los vectores de *Spacy* y **2)** transformadores de oraciones. Ambas aproximaciones son fundamentalmente diferentes en lo que corresponde al cálculo de esta métrica, la cual es simplemente el coseno del ángulo entre dos vectores (A, B) de álgebra lineal:

Ecuación 1. Coseno del ángulo de dos vectores (A, B) de álgebra lineal

$$\text{Similitud}\ (\text{A}, \text{B}) = \cos(\theta) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\|\|\vec{B}\|}$$

De manera análoga a la vectorización para los modelos SVM, el uso de los vectores de *Spacy* funciona a partir del cálculo de un vector promedio a partir de los vectores individuales de cada palabra, mientras que en el segundo caso, toda la oración está representada por un solo vector global. Intuitivamente, la segunda aproximación es más robusta que la primera, pues incorpora el sentido global de la oración de manera similar a como lo haría un ser humano. Esta afirmación no sólo es una intuición, sino que matemáticamente está respaldada, pues la representación vectorial en cada caso es diferente. En el primer caso, cada vector es de 300-dimensiones, mientras que en el segundo caso, cada vector es de 768-dimensiones, por lo que los transformadores de oraciones tienen más del doble de información que la primera alternativa. La siguiente figura resume dicha lógica:

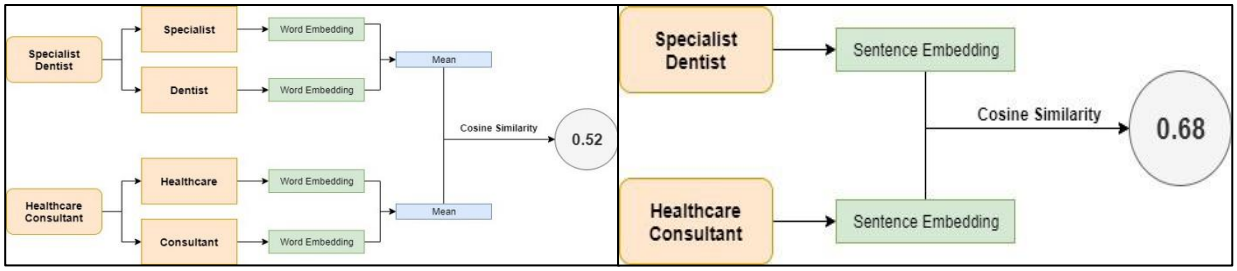


Ilustración 23.Diferencias en la similitud semántica para word embeddings y sentence embeddings, respectivamente

De la misma manera en que se muestra para el ejemplo anterior, es posible comparar las métricas de similitud entre diadas de oraciones para ambas alternativas. Para la alternativa de *word embeddings* se utilizaron los vectores de la oración preprocesada[13], mientras que para los transformadores se usaron las oraciones tal cual sin ningún tipo de preprocesamiento. En este caso, el transformador utilizado fue el mismo modelo BERT en español (BETO) (Canete, 2020), pero adaptado a la tarea específica de similitud de oraciones. En este caso, no se hizo ningún tipo de sobre ajuste de este, sino que se tomó el modelo original[14] pre-entrenado de *Hugging Face* (Hugging Face, 2021). Como ejemplo ilustrativo, la siguiente tabla resume dicha comparación para un árbol de problemas escogido arbitrariamente:

Problema: Insostenibilidad económica y social de las familias y mujeres Víctimas del Conflicto Armado		**Similitud Semántica**	
Tipo	*Oración*	*Word Embeddings*	*Transformers*
Causa	Mínimas oportunidades para la generación de ingresos para la población VCA	0.72	0.29
Causa	Deficiente aplicación de la Ley de Víctimas	0.59	0.31
Causa	Falta de apoyo de las entidades gubernamentales para la generación de ingresos de las familias y mujeres VCA	0.83	0.37
Causa	Mínimas iniciativas productivas	0.62	0.20
Efecto	Incremento de la población en condiciones de desplazamiento en los centros poblados	0.70	0.48
Efecto	Descomposición social	0.81	0.38
Efecto	Baja calidad de vida de las Víctimas del Conflito Armado	0.77	0.71
Efecto	Aumento de las Necesidades Básicas insatisfechas (NBI)	0.66	0.33

Tabla 27. Similitud semántica para un árbol de problemas arbitrario utilizando vectores de palabras y transformadores

Teniendo en cuenta que el modelo t*ransformer* es más robusto por todas las razones expuestas anteriormente, como parte de los entregables del proyecto, se realizó una visualización de estos vectores en un plano cartesiano utilizando como técnica de reducción de dimensionalidad

[13] Las mismas que se usaron para los modelos de clasificación (sin palabras vacías, signos de puntuación, caracteres extraños, etc.)
[14] hiiamsid/sentence_similarity_spanish_es

el t-Distributed Stochastic Neighbor Embedding (t-SNE) (Hinton, 2002) (Van der Maaten, 2008). Se escogió está técnica de reducción de dimensionalidad frente a otras aproximaciones más tradicionales como el análisis de componentes principales (PCA), dado que esta última alternativa es una técnica de reducción lineal que busca maximizar la varianza de las variables. En el caso particular del PLN, se sabe que las combinaciones de palabras (características) rara vez son lineales, por lo que se tomó el t-SNE como una alternativa más robusta para poder crear las visualizaciones a continuación.

La visualización propuesta es un aplicativo que contiene un gráfico de dispersión, donde cada punto representa una oración, el color indica el tipo de oración y el tamaño es proporcional a la similitud semántica con el respectivo problema. La siguiente figura es un ejemplo para el mismo árbol escogido anteriormente de manera arbitraria:

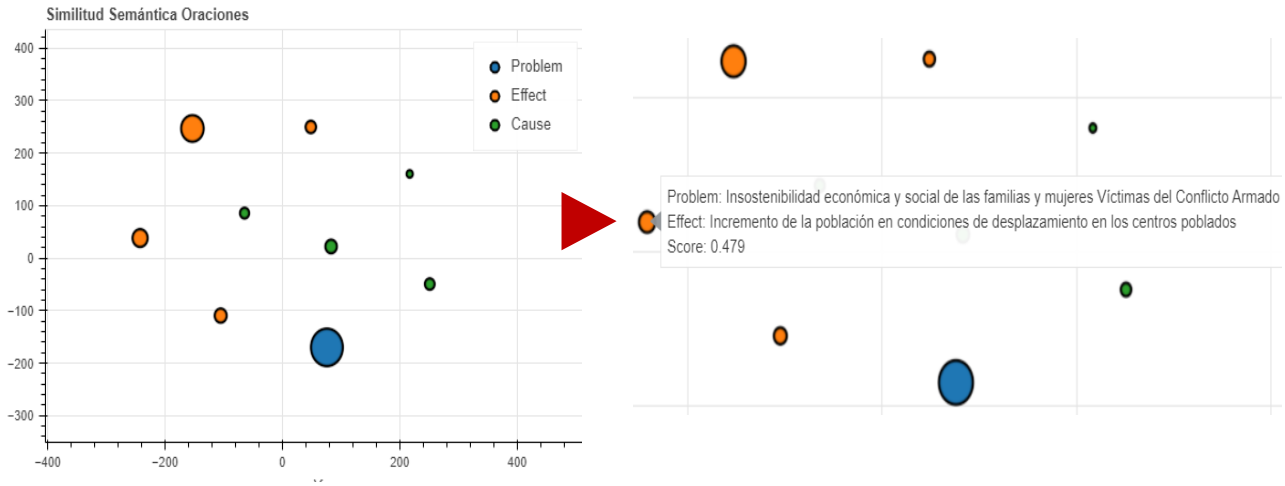
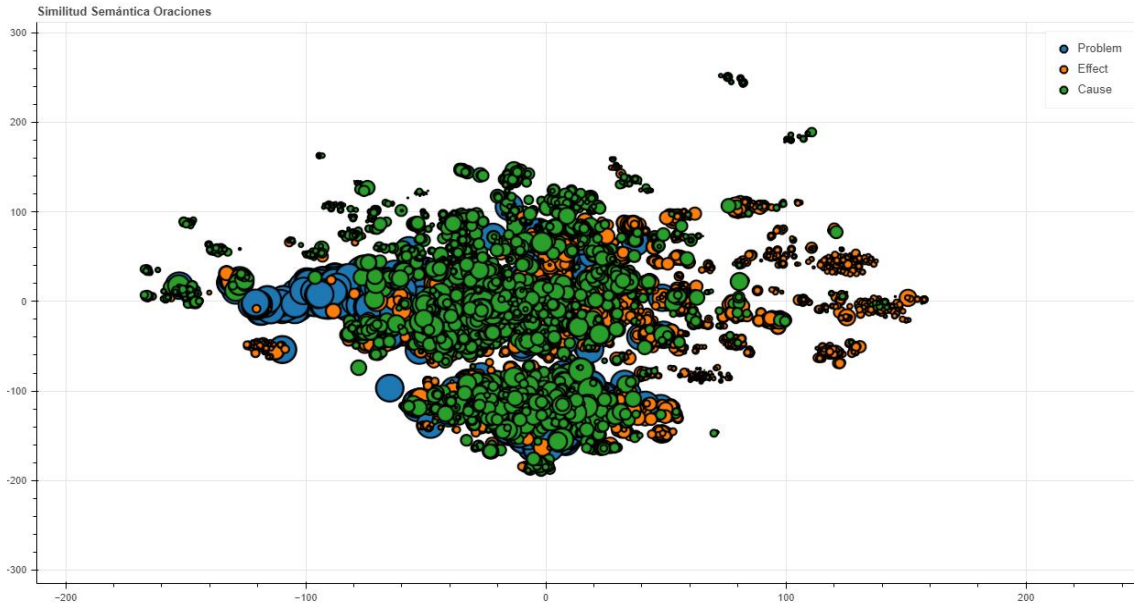


Ilustración 24. Similitud semántica de oraciones en un plano cartesiano (2-D) aplicando reducción de dimensionalidad con t-SNE para un árbol de problemas arbitrario

Esta visualización no está limitada al número de oraciones, e inclusive, es posible extenderla a todo el conjunto de datos:

Ilustración 25. Similitud semántica de oraciones en un plano cartesiano (2-D) aplicando reducción de dimensionalidad con t-SNE para todo el conjunto de datos



VIII. DEMOSTRACIÓN MODELOS ANALÍTICOS

Con el fin de mostrar cómo sería el ejercicio práctico de validación para un conjunto de oraciones reales candidatas a formar parte de un árbol de problemas. Se realizó dicha prueba con un árbol de problemas enfocado a temas de Analítica y *Big Data* perteneciente a CAOBA. Con los resultados del modelo BETO balanceado, se hizo una predicción por fuera de la muestra para estas oraciones, mostrándole al formulador del proyecto cual es la etiqueta sugerida en cada caso:

Tabla 28. Clasificación de oraciones para un árbol nuevo

Oración	Pred Label
Bajos niveles de digitalización de datos en empresas públicas y privadas	Problem
Capacidades insuficientes para la explotación de datos en organizaciones del sector público y privado del Distrito Capital	Problem
Baja apropiación y absorción de las tecnologías en Big Data y Analítica en las organizaciones del Distrito Capital	Cause
Bajo capital humano para la explotación de datos	Cause
Desconocimiento de las herramientas requeridas y procesos de aprestamiento en analítica de las organizaciones en el Distrito Capital	Cause
Dificultad para identificar necesidades y ejecutar proyectos en analítica que generen valor en las organizaciones del Distrito Capital	Cause
Falta de capacitación en metodologías y herramientas para el aprovechamiento de datos	Cause
Mala estimación de recursos necesarios para el desarrollo de proyectos de analítica y Big Data	Cause
Pocos proyectos de investigación e innovación en Big Data y Analítica	Cause
Difícil accesibilidad a soluciones de Big Data y Analítica ajustadas a las necesidades de las organizaciones	Effect
Fracasos en la implementación de proyectos para uso y explotación de datos	Effect
Rezago en el uso de soluciones y beneficios de analítica aplicados a servicios ciudadanos	Effect
Sobrecostos en la adquisición de bienes y servicios de Analítica y Big Data	Effect

Adicionalmente es posible utilizar el transformador de oraciones para calcular la similitud semántica entre los problemas sugeridos y las demás oraciones de causa/efecto:

Tabla 29. Similitud semántica para las oraciones y el problema "Bajos niveles de digitalización en empresas públicas y privadas"

	Problem: Bajos niveles de digitalización de datos en empresas públicas y privadas	Score
Cause	Bajo capital humano para la explotación de datos	0.58
Cause	Pocos proyectos de investigación e innovación en Big Data y Analítica	0.52
Cause	Baja apropiación y absorción de las tecnologías en Big Data y Analítica en las organizaciones del Distrito Capital	0.49
Cause	Falta de capacitación en metodologías y herramientas para el aprovechamiento de datos	0.49
Cause	Mala estimación de recursos necesarios para el desarrollo de proyectos de analítica y Big Data	0.44
Cause	Desconocimiento de las herramientas requeridas y procesos de aprestamiento en analítica de las organizaciones en el Distrito Capital	0.35
Cause	Dificultad para identificar necesidades y ejecutar proyectos en analítica que generen valor en las organizaciones del Distrito Capital	0.34
Effect	Fracasos en la implementación de proyectos para uso y explotación de datos	0.51
Effect	Difícil accesibilidad a soluciones de Big Data y Analítica ajustadas a las necesidades de las organizaciones	0.48
Effect	Sobrecostos en la adquisición de bienes y servicios de Analítica y Big Data	0.48
Effect	Rezago en el uso de soluciones y beneficios de analítica aplicados a servicios ciudadanos	0.39

Tabla 30. Similitud semántica para las oraciones y el problema "Capacidades insuficientes para la explotación de datos en organizaciones del sector público y privado del Distrito Capital"

	Problem: Capacidades insuficientes para la explotación de datos en organizaciones del sector público y privado del Distrito Capital	Score
Cause	Baja apropiación y absorción de las tecnologías en Big Data y Analítica en las organizaciones del Distrito Capital	0.66
Cause	Desconocimiento de las herramientas requeridas y procesos de aprestamiento en analítica de las organizaciones en el Distrito Capital	0.62
Cause	Dificultad para identificar necesidades y ejecutar proyectos en analítica que generen valor en las organizaciones del Distrito Capital	0.61
Cause	Falta de capacitación en metodologías y herramientas para el aprovechamiento de datos	0.55
Cause	Bajo capital humano para la explotación de datos	0.54
Cause	Mala estimación de recursos necesarios para el desarrollo de proyectos de analítica y Big Data	0.51
Cause	Pocos proyectos de investigación e innovación en Big Data y Analítica	0.43
Effect	Fracasos en la implementación de proyectos para uso y explotación de datos	0.54
Effect	Difícil accesibilidad a soluciones de Big Data y Analítica ajustadas a las necesidades de las organizaciones	0.53
Effect	Sobrecostos en la adquisición de bienes y servicios de Analítica y Big Data	0.41
Effect	Rezago en el uso de soluciones y beneficios de analítica aplicados a servicios ciudadanos	0.39

De la misma manera se aplicó el modelo sobre ajustado de entidades nombradas, pero para este caso particular no se reconoció ninguna etiqueta del conjunto definido anteriormente. Finalmente y de acuerdo con todo el análisis realizado hasta el momento se sabe que el desempeño del modelo de clasificación está supeditado al contexto de los proyectos de inversión sociales en áreas rurales de Cundinamarca, por lo que es de esperarse, que el modelo se vea limitado a la hora de clasificar otro tipo de proyectos (como el anterior) que estén fuera de este alcance, no obstante, esta demostración es solamente con propósitos ilustrativos.

IX. VALIDACIÓN CON MODELO DE ACEPTACIÓN TECNOLÓGICA (TAM)

Para revisar si el modelo tiene un impacto significativo en el negocio se aplica el Modelo de Aceptación Tecnológica (TAM[15]), el cual es uno de los modelos más empleados con éxito en la mayoría de las investigaciones. Este fue diseñado para predecir la aceptación de un sistema tecnológico dentro de una organización, en este caso se utilizó para validar el uso del modelo analítico por parte de los posibles usuarios que podrían llegar a utilizarlo en diferentes contextos.

El TAM tiene como objetivo explicar los factores que determinan el uso del sistema por cierto número de usuarios, sugiriendo que la utilidad y la facilidad de uso son determinantes en la intención de uso por parte del usuario, como se evidencia en la siguiente ilustración. La primera se refiere al grado en que una persona cree que usando un sistema en particular mejorará su desempeño en el trabajo, y la segunda señala hasta cual grado una persona cree que usando un sistema en particular realizará menos esfuerzo para desempeñar sus tareas.

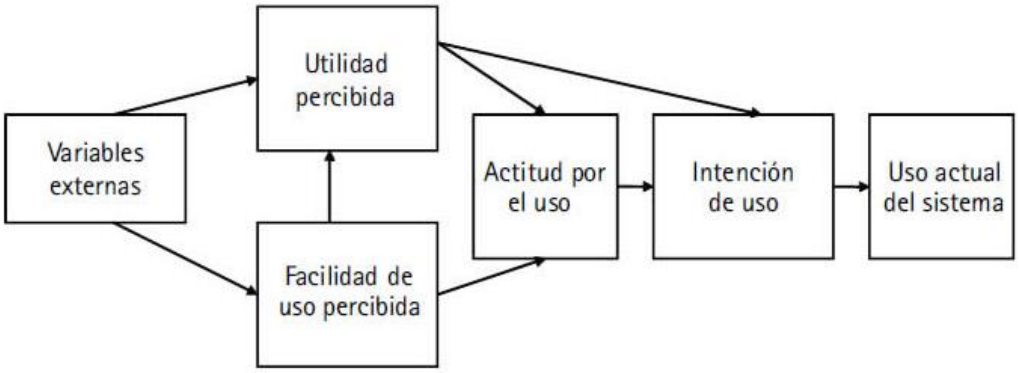


Ilustración 26. Estructura TAM

Se debe tener en cuenta que es ideal identificar variables externas que influyen de manera indirecta en estos frentes, las cuales determinan la relación que se tiene con el resultado del modelo analítico, por ejemplo, una de ellas es la intención de uso que tenga realmente el usuario (Yong, Rivas, & Chaparro, 2010). En la construcción del cuestionario para la evaluación de impacto al negocio, se utilizó la plantilla que se encuentra en la investigación realizada por (Pájaro Hernández, 2018).

[15] *Technology acceptance model.*

Se realizaron 3 sesiones personalizadas con 4 expertos en formulación de proyectos de la MML, en las cuales se explicó el objetivo del trabajo de grado y el por qué era necesario compartir los resultados con especialistas como ellos. Además, se hizo una demostración que permitió dar a entender el funcionamiento de los modelos y su uso práctico dentro del ejercicio de validación. Al final, se pidió el diligenciamiento de la encuesta que se encuentra en el anexo 7.

En cuanto a los resultados de este cuestionario, se encuentra que:

- Los expertos perciben que, con los resultados del modelo analítico, el formulador de proyectos podría ser más efectivo en su trabajo. Lo cual es bastante útil y costo-efectivo dentro del proceso de asesoría y formulación de proyectos.

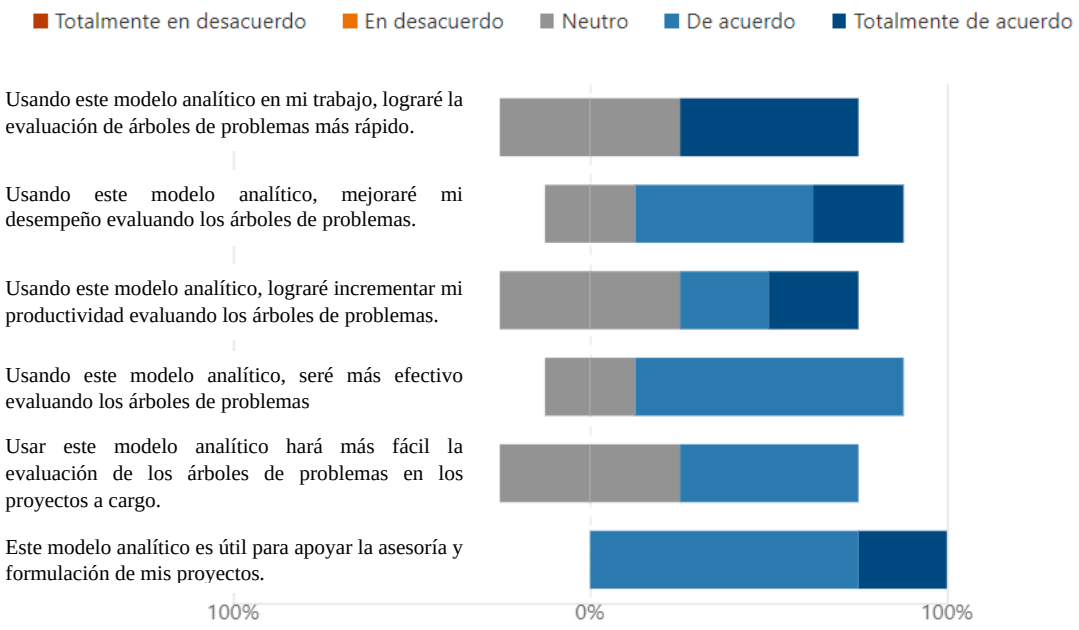
- Los expertos encuentran que será fácil hacer que el resultado del modelo analítico los ayude a evaluar árboles de problemas, puesto que la interacción con el mismo será sencilla y comprensible, permitiendo así que se pueda evaluar una gran cantidad de proyectos.

- Los expertos coinciden en que evaluar los árboles de problemas usando el resultado de este modelo analítico sí requerirá un esfuerzo mental. Además, no están seguros de que pueda ser fácil que un usuario se vuelva experto usando el modelo.

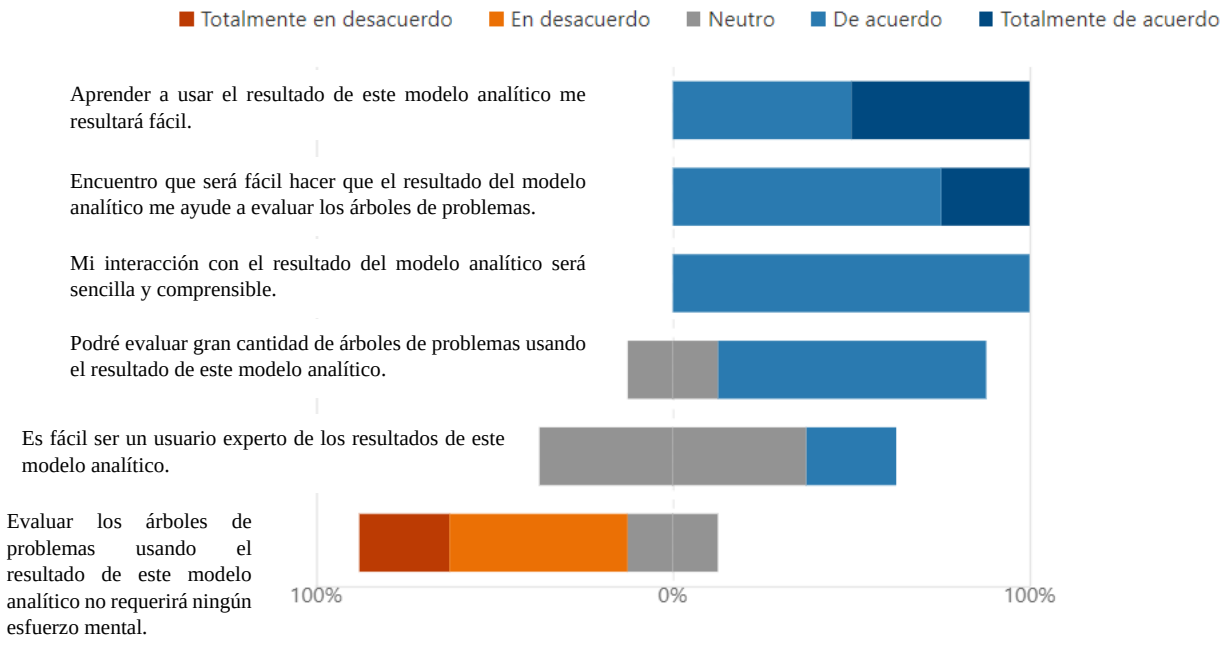
- Los expertos quieren usar los resultados del modelo analítico apenas puedan hacerlo y definitivamente lo recomendarían a otros colegas (*net promoting score* positivo).

Evaluación cuantitativa: A continuación, se muestran los resultados de la encuesta.

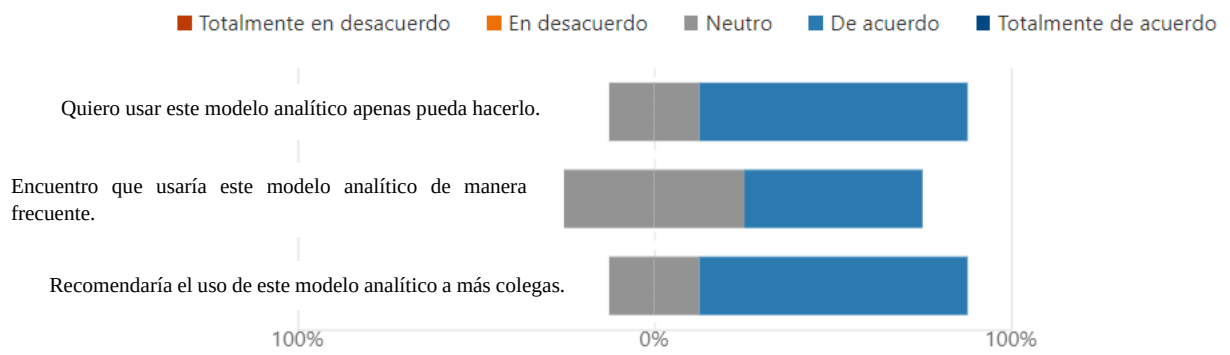
Sección 1. Se refiere a la posible mejora en el desempeño del trabajo apoyado por el modelo analítico propuesto.



Sección 2. Se refiere al esfuerzo en el uso del resultado del modelo analítico propuesto.



Sección 3. Se refiere a la percepción sobre la posibilidad de uso de este modelo analítico.



Evaluación cualitativa: Dentro de la retroalimentación cualitativa recibida por el panel de expertos se tocaron los siguientes puntos:

- En la práctica, uno de los principales retos a la hora de someter los proyectos, es la falta de conocimiento por parte de los responsables políticos en cada ente territorial. Capacitar a estas personas resulta costoso y difícil de llevar a la práctica, por lo que implementar esta solución analítica sería de gran ayuda para democratizar el conocimiento entorno a los árboles de problemas. No obstante, es importante reconocer que para que la solución sea exitosa, es imprescindible contar con un acompañamiento técnico en la parte de despliegue, dadas las barreras que existen en términos de adopción de tecnologías de la información e inclusive las implicaciones éticas detrás de estos proyectos de inversión.

- Otras de las inquietudes mencionadas por los expertos tienen que ver con los retos asociados a la formulación del árbol de problemas en sí, ya que evidentemente definir el problema central y las causas/efectos primarios y secundarios es la parte más difícil al principio del proceso. Por lo que esto inclusive podría tener más valor que la validación del árbol en sí. A pesar de que el objetivo principal del proyecto es la validación del árbol de problemas después de un trabajo de formulación previo, dentro de esta prueba de concepto sí se hace un avance en este tema con el modelo para el NER. Este último permitiría en un futuro vincular estas entidades a una ontología ya existente, de la tal manera que a partir de esta se puedan sugerir conceptos relacionados que apoyen la formulación del árbol. En el apartado de “Consideraciones Futuras” se hace una revisión más detallada de los *Knowledge Graphs* (KG) y cómo estos podrían aportar al ejercicio de formulación.

- Desde el punto de vista de la validación del árbol de problemas, una de las sugerencias fue tratar de incorporar de alguna manera las reglas existentes de los manuales de formulación como algo adicional dentro del proceso de validación. Por ejemplo: la redacción del problema central en estado negativo, no confundir el problema con la ausencia de una solución, etc. Adicionalmente se mencionó la gran utilidad que tendría desenmarañar los conceptos subyacentes que constituyen una oración como causa, problema o efecto. En la práctica esto es complicado, ya que esta rama de la analítica conocida como *Explainable AI* (XAI) con alternativas como SHAP (Lundberg, 2017), LIME (Ribeiro, 2016), ELI5 (Bonaros, 2021) y SKATER (Choudary, 2018) está muy bien documentada para los modelos clásicos de aprendizaje automático e inclusive algunos modelos de redes neuronales para la clasificación de imágenes. No obstante, la XAI tiene una serie de limitaciones conceptuales (Google, 2021) y sigue siendo una rama de la analítica en desarrollo, donde modelos como los *transformer* se encuentran aún dentro de la frontera del conocimiento. En síntesis, el ejercicio del TAM arroja que esta aproximación desde la XAI es relevante para el proceso de validación, por lo que en trabajos futuros podría abordarse.

- La percepción por parte de algunos expertos fue que los modelos desconocen muchas de las realidades particulares de las comunidades por lo que ciertas jerarquizaciones, por ejemplo, en el caso de la similitud semántica arrojan validaciones que no son acertadas. En este caso, se recalcó el hecho de que estos modelos no pretenden reemplazar el conocimiento de causa de la persona que está formulando el proyecto y que está jerarquización es netamente desde la construcción semántica de cada oración (causa o efecto) en relación con el problema. Lo que se pretende en este caso, es tener una medida objetiva que permita ayudar a “centrar el análisis de causas y efectos entorno a un solo problema central. Lo que permite acotar el análisis y ser más efectivo en recomendar soluciones” (Pájaro Hernández, 2018).

- Una de las limitantes viene inherentemente desde el sesgo que tienen los modelos de aprendizaje automático con relación a la base de datos con la que fueron entrenados. En este caso hay tres tipos de sesgo presentes: sesgo de medición, sesgo de muestra y sesgos de prejuicio. Siendo los dos primeros los que más llamaron la atención de los expertos.

- Sesgo de medición: A juicio de los expertos, el árbol de problemas real y aprobado usado en la demostración tenía algunas oraciones redactadas incorrectamente, problemas con soluciones triviales, efectos y causas no relacionados con el problema, entre otros. Esto fue una muestra en vivo de porque la tarea de construcción de los árboles de problemas es complicada, dado su alcance multidisciplinario lo cual hace difícil llegar a un conceso. Aun cuando en este trabajo de grado se hace el supuesto de que este tipo de sesgo es mínimo al ser proyectos reales y aprobados, esto sigue siendo un supuesto fuerte tal como lo arrojo el ejercicio del TAM.

- Sesgo de la muestra: En segundo lugar, el tipo de proyectos usado como insumo del modelo generó bastantes inquietudes durante el ejercicio. Este tipo de sesgo ya estaba previsto *ex-ante* al TAM[16], pues ya se sabía que el modelo estaba acotado a proyectos de inversión muy específicos (educación, infraestructura, transporte, etc.) en municipios rurales de Cundinamarca. Por lo que se hizo la salvedad de que en la etapa del despliegue esto iba a ser uno de los mayores retos, pues el modelo no es (aún) capaz de generalizar cualquier tipo de proyecto. Sin embargo, la prueba de concepto arroja que esto sería factible en un futuro.

- Sesgos de prejuicio: No fue una inquietud dentro del ejercicio del TAM, no obstante, es importante resaltarlo también. Este tipo de sesgo ocurre cuando el algoritmo aprende prejuicios o estereotipos erróneos que no reflejan la realidad de la sociedad. En el caso puntual del conjunto de datos usado, esto puede reflejarse en estereotipos de comunidades especificas siendo asociadas con: violencia, corrupción, carentes de educación, poco desarrolladas, o inclusive, los prejuicios de los mismos formuladores. Lo cual hace parte en cierta medida de las implicaciones éticas que hay que tener en cuenta a la hora de implementar estos modelos.

X. CONSIDERACIONES FUTURAS

Este trabajo tiene como alcance principal la clasificación de oraciones con un modelo de aprendizaje automático robusto para diferentes aproximaciones sugeridas por la literatura, la

[16] Tal como se mostró en la predicción del árbol de *Big Data* y Analítica de CAOBA.

creación de un modelo de detección de entidades nombradas personalizado acorde al contexto con el que se está trabajando y un modelo para calcular el grado de similitud semántica entre diadas de oraciones. Dentro de la tarea de validar los árboles de problemas existen otras alternativas que permiten enriquecer este proceso, por lo que es ideal que en trabajos futuros se implemente el uso de *knowledge graphs* (KG).

El término KG fue introducido por Google en 2012 para referirse a su base de conocimiento principal. Hoy en día, los KG se utilizan ampliamente en cualquier cosa, desde motores de búsqueda y *chatbots*, hasta recomendadores de productos y sistemas autónomos. De acuerdo con (IBM, 2021) se entiende a los KG como una red semántica que representa entidades del mundo real, es decir, objetos, eventos, situaciones o conceptos, e ilustra la relación entre ellos. La cual generalmente se almacena y visualiza como un grafo.

Los KG hacen parte de los algoritmos de aprendizaje automático para PLN que construyen una vista integral de nodos, bordes y etiquetas a través de un proceso llamado enriquecimiento semántico. Cuando se ingieren datos, este proceso permite que los KG identifiquen objetos individuales y comprendan las relaciones entre los diferentes objetos. Luego, este conocimiento práctico se compara e integra con otros conjuntos de datos, que son relevantes y de naturaleza similar. Una vez que se completa un KG, esto permite que los sistemas de búsqueda y respuesta a preguntas, recuperen y reutilicen respuestas completas a consultas determinadas. Los esfuerzos de integración de datos en torno a los KG también pueden respaldar la creación de nuevos conocimientos, estableciendo conexiones entre puntos de datos que pueden no haberse realizado antes.

Un KG organiza e integra datos de acuerdo con un conocimiento y aplica un razonador para obtener nuevos. Vale la pena aclarar que teniendo en cuenta varias investigaciones base, se encuentra que un KG no es diferente de una base de conocimiento u ontología (IBM, 2021). Algunas oportunidades en las cuales se puede aprovechar el uso de KG son: **1)** aumentar la base de datos cuando estos son insuficientes; **2)** mejorar la distinción de clases cuando no existan datos de entrenamiento suficientes; y, **3)** mejorar el proceso de toma de decisiones al asignar explicaciones a las predicciones realizadas en algunos nodos (Chowdhury, 2019).

A continuación, se muestra un ejemplo de KG. De la oración que se muestra a la izquierda en la siguiente ilustración, se pueden extraer las entidades: ‘Albert Einstein’, ‘Alemania’, ‘Físico Teórico’ y ‘Teoría de la Relatividad’; y las relaciones: ‘nacido en’, ‘con ocupación’ y ‘desarrolló’. Una vez que este fragmento del KG se incorpora en un grafo más grande, se obtienen enlaces adicionales (mostrados por bordes punteados) como: ‘un físico teórico es una especie de físico que practica la física’, y que ‘la teoría de la relatividad es una rama de la física’. (Stanford, s.f.)

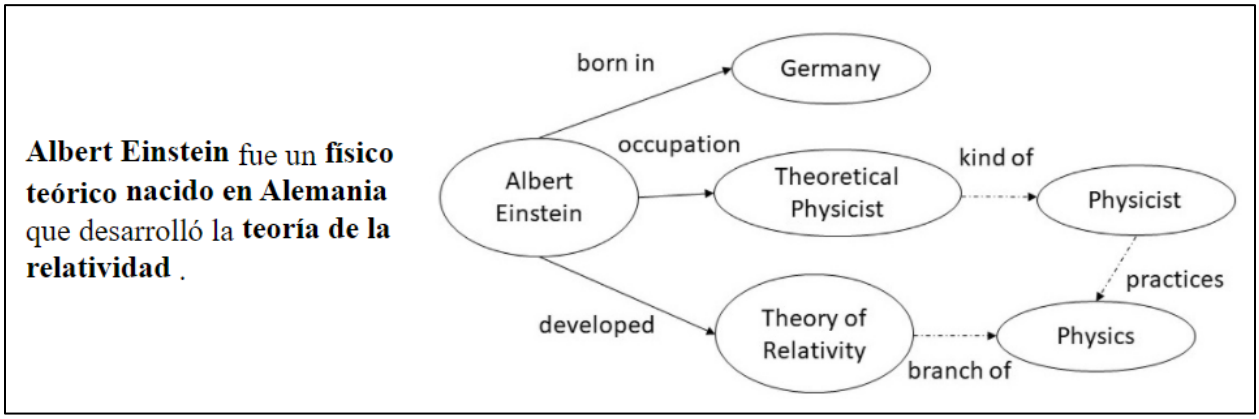


Ilustración 27. Ejemplo de KG

Los casos más comunes de KG son el de Google para la búsqueda web, o el de Amazon para la categorización de sus productos, e inclusive, los motores en redes sociales como Facebook. Otros que están abiertamente disponibles son: *DBpedia*, *Wikidata*, *WordNet* o *Geonames*, entre otros (The Alan Turing Institute, s.f.). Por otra parte, se encuentra que una de las herramientas más utilizadas para la manipulación de KG es *Neo4J*, pues es uno de los mejores motores en el mercado para interactuar con grafos de este tipo (Neo4J, s.f.).

Resulta importante discernir qué es y qué no es un KG. Por ejemplo, los conjuntos de datos estadísticos del Producto Interno Bruto (PIB) de los países no son un grafo, pues no existen conexiones que permitan formar una red. De manera similar, no todas las bases de conocimiento pueden ser ejemplos de KG, ya que estas necesitan que las descripciones de entidades estén interconectadas entre sí, es decir que aquellas bases sin estructura formal ni semántica son ejemplos de KG, como lo puede ser una base de preguntas sobre un producto de software (Ontotext, s.f.).

Una vez entendido el alcance de los KG, se puede afirmar que el uso de estos grafos es coherente con el objetivo de negocio que tiene IGG para validar los árboles de problemas. Esto debido a que los KG demuestran su capacidad para ahorrar tiempo, eliminar la recopilación manual de datos (como es el caso de la validación subjetiva de los proyectos) y el trabajo de integración para respaldar la toma de decisiones de inversión.

Como una posible alternativa, se recomienda utilizar la ontología *Sustainable Development Goals Interface* (SDGI) ya que su aplicación puede ser idónea en este caso, puesto que provee representaciones semánticamente coherentes de entidades relevantes y que tienen relación con los objetivos de desarrollo sostenible (ODS) (OLS Ontology Searcg, s.f.). Los proyectos de inversión social sometidos bajo la MML van en línea con el alcance que tiene esta ontología.

XI. CONCLUSIONES

El proyecto desarrollado en conjunto con IGG tiene como objetivo desarrollar una prueba de concepto para realizar la validación de árboles de problemas formulados bajo la MML a través de técnicas basadas en el PLN. Esta tarea es relevante dentro del marco del diseño de políticas públicas, pues el proceso de validación es difícil y costoso, dados los elementos transdisciplinarios y sistémicos que involucran a profesionales de múltiples disciplinas, así como intereses particulares. Para lograr este objetivo general, se utilizó un repositorio de árboles de problemas correspondiente a proyectos de inversión reales y aprobados por el DNP, los cuales estaban etiquetados como causa, problema y efecto. A partir de este conjunto de datos, se plantearon tres objetivos específicos para la minería de datos correspondientes a las tareas de: clasificación de oraciones, NER y similitud semántica.

En primer lugar para el problema de clasificación se utilizó la base de datos mencionada anteriormente complementada con oraciones *dummy* recopiladas a partir de las entradas de Wikipedia de varios municipios de Cundinamarca y un repositorio con nombres de proyectos ejecutados por el DNP. Se probaron diferentes modelos basados en varios paradigmas propuestos por la literatura: BOW, *word embeddings* y modelos de lenguaje *transformer*. A partir de esta fase de experimentación se propone como modelo, la utilización de un modelo BERT en español (BETO) entrenado a partir de un conjunto de datos balanceado, dado que tuvo métricas superiores en el conjunto de prueba aún con poco preprocesamiento de las oraciones. Adicionalmente, se concluye que no existen diferencias significativas entre los modelos más sencillos (BOW) y los

más sofisticados (*embeddings* & *transformer*). Esto último, debido a que se contaba con un *dataset* con proyectos reales y aprobados en un contexto similar, facilitando así la detección de patrones que permitieran discernir entre las diferentes clases. En un futuro, si se desea llegar a un modelo de clasificación más generalizable es necesario enriquecer la base de datos con más temas de proyectos o pensar en alternativas como *Low-shot* o *Zero-shot Learning* (Yan, 2018), dada la poca disponibilidad de *datasets* anotados

En segundo lugar para el modelo NER se entrenó un modelo sobre ajustado FLERT (Schweter, Flert: Document-level features for named entity recognition., 2020), el cual es capaz de reconocer principalmente municipios, instituciones educativas, departamentos y veredas. Esto con el fin de que dentro de un futuro se puedan vincular estas entidades a una ontología existente y a partir de ahí sugerir conceptos adicionales que permitan enriquecer la construcción del árbol de problemas a partir de un KG.

En tercer lugar se plantea un modelo basado en transformadores de oraciones para calcular la similitud semántica entre diadas de oraciones problema y causa/efecto. Lo que se pretende en este caso, es tener una medida objetiva que permita ayudar a "centrar el análisis de causas y efectos entorno a un solo problema central. Lo que permite acotar el análisis y ser más efectivo en recomendar soluciones" (Pájaro Hernández, 2018)

Finalmente se practicó un TAM con la ayuda de un panel de expertos en formulación de proyectos sociales bajo la MML. El objetivo era medir cuantitativa y cualitativamente la utilidad de los modelos analíticos propuestos sobre el negocio. De este ejercicio se obtienen los siguientes hallazgos y recomendaciones:

- Los expertos consideran que podrían ser más efectivos al usar avances como estos en su día a día y creen posible que la interacción sea fácil y comprensible para ellos, lo cual sería de gran utilidad dentro del proceso general de sometimiento de proyectos bajo la MML. Sin embargo, sienten que puede existir un gran esfuerzo mental al usarlo. Por otra parte, quisieran que se implemente en un plazo corto de tiempo ya que lo ven bastante provechoso para ellos, puesto que actualmente existen limitaciones en cuanto a las personas que tienen el conocimiento para la evaluación de árboles de problemas y soluciones como esta pueden ser una gran ayuda para democratizarlo. Aunque, vale la pena aclarar que esta herramienta no es un reemplazo del experto, es un apoyo para el mismo, de manera tal que sus tareas

diarias puedan ser más llevaderas y así se evalúen más proyectos, lo cual se traduce en algo positivo para la sociedad.

- Desde el punto de vista de los expertos y en línea con las propuestas para trabajos futuros se tienen como oportunidades: **1)** el uso de modelos analíticos para la formulación de árboles de problemas (por ejemplo, a partir del uso de KG) y **2)** la posibilidad de abordar la validación de los mismos desde un enfoque basado en XAI para explorar los conceptos subyacentes que componen un efecto, problema o causa dentro de la clasificación del árbol de problemas.

XII. BIBLIOGRAFÍA

Ajitesh, K. (4 de Septiembre de 2020). *Vitalflux*. Obtenido de https://vitalflux.com/micro-average-macro-average-scoring-metrics-multi-class-classification-python/

Amat, J. (Abril de 2017). *Máquinas de Vector Soporte (Support Vector Machines, SVMs)*. Obtenido de https://www.cienciadedatos.net/documentos/34_maquinas_de_vector_soporte_support_vector_machines

Bojanowski, P. G. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 135-146.

Bonaros, B. (12 de Julio de 2021). *Towards Data Science*. Obtenido de Debug Models And Explain Predictions Using Eli5: https://towardsdatascience.com/debug-models-and-explain-predictions-using-eli5-9f856ae74d16

Canete, J. C. (2020). Spanish pre-trained bert model and evaluation data. *Pml4dc at iclr, 2020*.

Cañete, J. (19 de Octubre de 2021). *Spanish Word Embeddings*. Obtenido de GitHub: https://github.com/dccuchile/spanish-word-embeddings#fasttext-embeddings-from-sbwc

Chauhan, A. (3 de Mayo de 2020). *Medium*. Obtenido de Training Custom NER Model Using Flair: https://medium.com/thecyphy/training-custom-ner-model-using-flair-df1f9ea9c762

Choudary, P. (22 de Marzo de 2018). *O'Reilly*. Obtenido de Interpreting predictive models with Skater: Unboxing model opacity: https://www.oreilly.com/content/interpreting-predictive-models-with-skater-unboxing-model-opacity/

Chowdhury, S. (18 de 7 de 2019). *Towards Data Science*. Obtenido de https://towardsdatascience.com/knowledge-graph-bb78055a7884

Conneau, A. K. (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Departamento Administrativo Nacional de Estadística. (11 de 10 de 2021). *Departamentos y municipios de Colombia*. Obtenido de Stio web Datos abiertos: https://www.datos.gov.co/Mapas-Nacionales/Departamentos-y-municipios-de-Colombia/xdk5-pm3f

Departamento Nacional de Planeación. (19 de octubre de 2021). *DNP-LocalizacionProyecto*. Obtenido de Datos abiertos: https://www.datos.gov.co/Econom-a-y-Finanzas/DNP-LocalizacionProyecto/xikz-44ja

Devlin, J. C. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Devlin, J. C. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Devlin, J. C. (2018). Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Explosion. (28 de Septiembre de 2021). *Language Processing Pipelines*. Obtenido de Spacy: https://spacy.io/usage/processing-pipelines

Gil, C. (Junio de 2018). *RPubs*. Obtenido de https://rpubs.com/Cristina_Gil/SVM

Google. (18 de Noviembre de 2021). *Introduction to Vertex Explainable AI* . Obtenido de AI Explanations Whitepaper: https://storage.googleapis.com/cloud-ai-whitepapers/AI%20Explainability%20Whitepaper.pdf

Hinton, G. &. (2002). Stochastic neighbor embedding. *NIPS Vol. 15*, 833-840.

Hugging Face. (11 de Noviembre de 2021). *hiiamsid/sentence_similarity_spanish_es*. Obtenido de Hugging Face: https://huggingface.co/hiiamsid/sentence_similarity_spanish_es

Hugging Face. (31 de octubre de 2021). *Hugging Face*. Obtenido de Transformers: https://huggingface.co/transformers/index.html

Hugging Face. (10 de noviembre de 2021). *Hugging Face*. Obtenido de Models: https://huggingface.co/models?language=es&pipeline_tag=text-classification&sort=downloads

IBM. (12 de 4 de 2021). Obtenido de https://www.ibm.com/cloud/learn/knowledge-graph

Infraestructura de Datos Espaciales Cundinamarca IDEC. (15 de 10 de 2021). *Veredas. Departamento de Cundinamarca*. Obtenido de Mapas y estadísticas: https://mapas.cundinamarca.gov.co/datasets/d6eacb73875d4ae29ecb3bd468590044_2/about

Inversiones Gutiérrez García. (20 de Agosto de 2021). *Nosotros: Inversiones Gutiérrez García*. Obtenido de Página web Inversiones Gutiérrez García: http://www.igg.com.co

Keras. (5 de noviembre de 2021). *Keras*. Obtenido de Embedding layer: https://keras.io/api/layers/core_layers/embedding/

Kowsari, K. J. (2019). Text classification algorithms: A survey. *Information, 10(4)*.

Liu, Y. O. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lundberg, S. M. (2017). A unified approach to interpreting model predictions. *Proceedings of the 31st international conference on neural information processing systems*, 4768-4777.

Mihaela, M. (2019). *Efficiency of SVM classifier with Word2Vec and Doc2Vec models.* Sciendo.

Mikolov, T. C. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM, 38(11)*, 39-41.

Ministerio de Educación Nacional. (16 de 10 de 2021). *Ministerio de Educación Nacional*. Obtenido de Buscando Colegio: https://sineb.mineducacion.gov.co/bcol/app

Neo4J. (s.f.). Obtenido de https://neo4j.com/developer/graph-data-science/build-knowledge-graph-nlp-ontologies/

Nothman, J., Ringland, N., Radford, W., Murphy, T., & Curran, J. R. (2017). Learning multilingual named entity recognition from Wikipedia.

OLS Ontology Searcg. (s.f.). Obtenido de https://www.ebi.ac.uk/ols/ontologies/sdgio

Ontotext. (s.f.). Obtenido de https://www.ontotext.com/knowledgehub/fundamentals/what-is-a-knowledge-graph/

Ortegón, E., Pacheco , J., & Prieto, A. (2005). *Metodología del marco lógico para la planificación, el seguimiento y la evaluación de proyectos y programas.* Santiago de Chile.: Naciones Unidas. CEPAL.

Pájaro Hernández, J. P. (2018). Procesamiento de Lenguaje Natural para la evaluación de problemas sociales. Bogotá, Colombia: Pontificia Universidad Javeriana.

Pedregosa, F. V. (2011). Scikit-learn: Machine learning in Python. *The Journal of machine Learning research, 12*, 2825-2830.

Pennington, J. S. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532-1543.

Pyspellchecker. (8 de Noviembre de 2021). *Pyspellchecker*. Obtenido de Pyspellchecker: https://pyspellchecker.readthedocs.io/en/latest/

Ribeiro, M. T. (2016). "Why should i trust you?" Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 1135-1144.

Schweter, S. &. (2020). Flert: Document-level features for named entity recognition. *arXiv preprint arXiv:2011.06993*.

Schweter, S. &. (2 de Octubre de 2021). *flairNLP / flair*. Obtenido de GitHub: https://github.com/flairNLP/flair/blob/master/resources/docs/TUTORIAL_7_TRAINING_A_MODEL.md

Scikit learn. (7 de noviembre de 2021). *Scikit learn*. Obtenido de sklearn.svm.SVC: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC

Scikit-learn. (4 de noviembre de 2021). *Scikit-learn*. Obtenido de Scikit-learn Machine Learning in Python: https://scikit-learn.org/stable/index.html

Stanford. (s.f.). Obtenido de https://web.stanford.edu/class/cs520/2020/notes/What_is_a_Knowledge_Graph.html

Taulé, M. M. (2008). Ancora: Multilevel Annotated Corpora for Catalan and Spanish. *Proceedings of 6th International Conference on Language Resources and Evaluation*.

Terechshenko, Z. L. (2021). A Comparison of Methods in Political Science Text Classification: Transfer Learning Language Models for Politics. *SSRN*, 5-11.

The Alan Turing Institute. (s.f.). Obtenido de https://www.turing.ac.uk/research/interest-groups/knowledge-graphs

Treleaven, B. B. (2014). Social media analytics: a survey of techniques, tools and platforms. *AI Soc*, 89–116.

Van der Maaten, L. &. (2008). Visualizing data using t-SNE. *Journal of machine learning research, 9(11)*.

Wang, A. S. (2018). Glue: A multitask benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Wei, J. &. (2019). Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Wikipedia. (28 de Septiembre de 2021). *Bojacá*. Obtenido de Wikipedia: https://es.wikipedia.org/wiki/Bojac%C3%A1

Wirth, R. &. (2000). CRISP-DM: Towards a standard process model for data mining. *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining (Vol. 1)*. Londres: Springer-Verlag.

Wolf, T. C. (2020). Transformers: State-of-the-art natural language processing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38-45.

Yan, L. Z. (2018). Few-shot learning for short text classification. *Multimedia Tools and Applications, 77(22)*, 29799-29810.

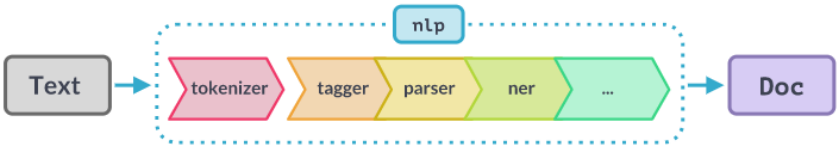
Yang, Z. D. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*.

Yin, W. K. (2017). Comparative study of CNN and RNN for natural language processing. *arXiv preprint arXiv:1702.01923*.

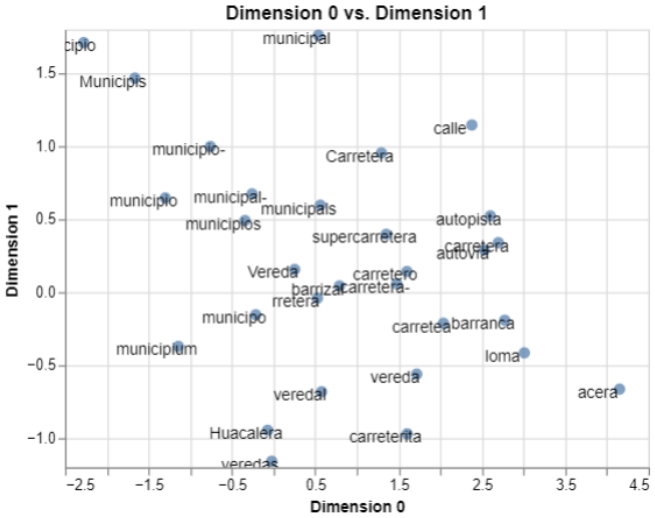
Yong, L., Rivas, L., & Chaparro, J. (Abril de 2010). *SciELO*. Obtenido de http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0121-50512010000100014

XIII. ANEXOS

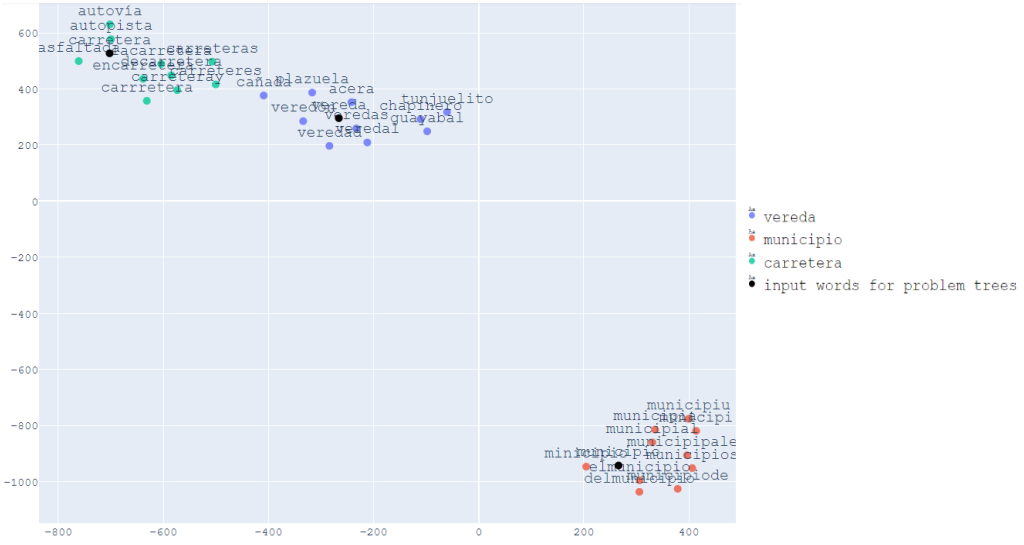
Anexo 1. Pipeline para el PLN de Spacy



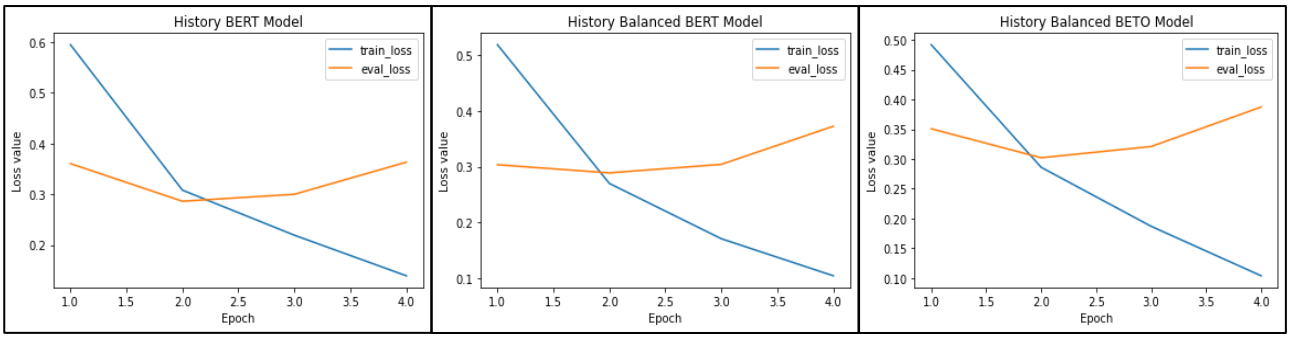
Anexo 2. Vectores para un subconjunto de palabras con el paquete Spacy



Anexo 3. Vectores (Cañete, 2021) provenientes del SUC para un subconjunto de palabras arbitrarias



Anexo 4. Historial de entrenamiento para 4 épocas de los modelos BERT



Anexo 5. Ejemplo para el formato de etiquetado de las oraciones

```
George N B-PER
Washington N I-PER
went V O
to P O
Washington N B-LOC

Sam N B-PER
Houston N I-PER
stayed V O
home N O
```

Anexo 6. Comparación de (Canete, 2020) entre el modelo BETO y el modelo BERT multilingüe para varias tareas del PLN

Task	BETO-cased	BETO-uncased	Best Multilingual BERT	Other results
POS	98.97	98.44	97.10 [2]	98.91 [6], 96.71 [3]
NER-C	**88.43**	82.67	87.38 [2]	87.18 [3]
MLDoc	95.60	**96.12**	95.70 [2]	88.75 [4]
PAWS-X	89.05	89.55	90.70 [8]	
XNLI	**82.01**	80.15	78.50 [2]	80.80 [5], 77.80 [1], 73.15 [4]

Anexo 7. Cuestionario TAM del proyecto de analítica

Objetivo: Evaluar la utilidad percibida al presentar un modelo analítico para clasificación de árboles de problemas para la formulación de proyectos sociales, cuya implementación podría impactar de manera relevante al negocio.

1. Utilidad percibida: Se refiere a la posible mejora en el desempeño del trabajo apoyado por el modelo analítico propuesto.

Pregunta / Respuesta	Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo
Usando este modelo analítico en mi trabajo, lograré la evaluación de árboles de problemas más rápido.					
Usando este modelo analítico, mejoraré mi desempeño evaluando los árboles de problemas.					
Usando este modelo analítico, lograré incrementar mi productividad evaluando los árboles de problemas.					
Usando este modelo analítico, seré más efectivo evaluando los árboles de problemas					
Usar este modelo analítico hará más fácil la evaluación de los árboles de problemas en los proyectos a cargo.					
Este modelo analítico es útil para apoyar la asesoría y formulación de mis proyectos.					

2. Facilidad de uso percibida: Se refiere al esfuerzo en el uso del resultado del modelo analítico propuesto.

Pregunta / Respuesta	Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo
Aprender a usar el resultado de este modelo analítico me resultará fácil.					
Encuentro que será fácil hacer que el resultado del modelo analítico me ayude a evaluar los árboles de problemas.					
Mi interacción con el resultado del modelo analítico será sencilla y comprensible.					
Podré evaluar gran cantidad de árboles de problemas usando el resultado de este modelo analítico.					
Es fácil ser un usuario experto de los resultados de este modelo analítico.					
Evaluar los arboles de problemas usando el resultado de este modelo analítico no requerirá ningún esfuerzo mental.					

3. Intención de uso percibida: Se refiere a la percepción sobre la posibilidad de uso de este modelo analítico.

Pregunta / Respuesta	Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo
Quiero usar este modelo analítico a penas pueda hacerlo.					
Encuentro que usaría este modelo analítico de manera frecuente.					
Recomendaría el uso de este modelo analítico a más colegas.					