PROBLMEA 2, DOCUMENTACIÓN: PRUEBA TÉCNICA PARA AI DEVELOPER JACOB. T.

Propuesta usando Visión artificial y NLP

Índice

Indice de Figuras

Introducción.

La automatización en la clasificación de solicitudes es un campo emergente dentro de la inteligencia artificial, específicamente en el ámbito del procesamiento del lenguaje natural (NLP). Este enfoque busca optimizar procesos que tradicionalmente requieren intervención humana, como la categorización, priorización y análisis de grandes volúmenes de datos textuales. La capacidad de comprender y procesar solicitudes de manera eficiente es fundamental en sectores como atención al cliente, gestión de tickets de soporte y administración de documentos.

El objetivo principal de este proyecto es desarrollar un sistema automatizado capaz de clasificar solicitudes textuales en diferentes categorías de manera precisa y eficiente. Para lograr esto, se emplean técnicas de NLP combinadas con modelos de aprendizaje profundo, como redes neuronales recurrentes (RNN) y transformadores, que han demostrado ser efectivos en tareas de comprensión del lenguaje natural. Estas tecnologías permiten analizar el contenido textual de las solicitudes, identificar patrones relevantes y asignarlas a categorías predefinidas, optimizando así los tiempos de respuesta y la calidad del servicio.

Este tipo de sistemas tiene aplicaciones en diversos campos, tales como:

- Atención al cliente: Clasificación automática de correos electrónicos, mensajes o tickets de soporte técnico para derivarlos al departamento correspondiente.
- Gestión documental: Organización y categorización de documentos en bases de datos empresariales.
- Monitoreo de redes sociales: Análisis y clasificación de comentarios o menciones para detectar tendencias, quejas o requerimientos de usuarios.
- Administración pública: Priorización de solicitudes ciudadanas en función de su urgencia o relevancia.

Para el sistema desarrollado se pueden realizar las siguientes tareas clave a futuro:

1. Preprocesamiento del texto: Limpieza y normalización de los datos textuales para garantizar un análisis eficiente.
2. Clasificación automática: Uso de modelos entrenados para identificar la categoría más adecuada para cada solicitud.
3. Análisis de prioridad: Asignación de niveles de urgencia basados en palabras clave, tono del mensaje y contexto.
4. Generación de reportes: Creación de registros detallados en formato estructurado (por ejemplo, JSON, CSV, TXT) para facilitar la integración con otros sistemas.

La implementación de este sistema no solo reduce la carga operativa en tareas repetitivas, sino que también mejora la precisión y consistencia en la clasificación de solicitudes. En este proyecto, se emplean herramientas de software como Python y bibliotecas especializadas en NLP, tales como SpaCy, NLTK y Hugging Face Transformers, para construir un modelo robusto de clasificación. Además, se integran técnicas de vectorización de texto, como TF-IDF y embeddings, que permiten capturar tanto el significado semántico como las relaciones contextuales entre palabras.

El desarrollo de este sistema busca sentar las bases para futuras implementaciones en diferentes secciones de la industria, donde la automatización de la clasificación de solicitudes pueda escalarse y adaptarse a contextos específicos. De esta manera, se contribuye a la transformación digital de procesos críticos, alineándose con las demandas de un entorno cada vez más orientado hacia la eficiencia y la innovación tecnológica.

Requisitos Técnicos

Para la implementación del sistema de automatización de clasificación de solicitudes, es necesario cumplir con ciertos requisitos técnicos que garantizan un funcionamiento óptimo. A continuación, se describen las herramientas, bibliotecas y configuraciones necesarias para desarrollar y ejecutar el proyecto:

1. Herramientas de Software

1. Python 3.12 o superior: Lenguaje de programación utilizado para el desarrollo del sistema, debido a su flexibilidad y amplia disponibilidad de bibliotecas para procesamiento del lenguaje natural (NLP).
2. Gestor de paquetes pip: Herramienta para instalar y administrar las bibliotecas requeridas en el entorno de desarrollo.
3. Jupyter Notebook o IDE equivalente: Para la ejecución interactiva y depuración del código durante el desarrollo del sistema, puede ser Visual Studio o el IDE de su preferenica.

2. Bibliotecas y Dependencias

El sistema requiere la instalación de las siguientes bibliotecas y dependencias para llevar a cabo las tareas de procesamiento de texto y clasificación:

- SpaCy: Biblioteca para el procesamiento eficiente de texto, que incluye herramientas para tokenización, lematización y análisis sintáctico.
- NLTK (Natural Language Toolkit): Conjunto de herramientas para realizar tareas de preprocesamiento de texto, como eliminación de stopwords y análisis gramatical.
- Hugging Face Transformers: Para la implementación de modelos avanzados de NLP, como BERT y GPT, que permiten capturar relaciones semánticas en el texto.
- Scikit-learn: Biblioteca para la implementación de algoritmos de clasificación, vectorización de texto (TF-IDF) y evaluación de modelos.
- Pandas: Para la manipulación y análisis de datos tabulares, facilitando la organización de solicitudes y resultados.
- NumPy: Para realizar operaciones matemáticas avanzadas y manipulación de matrices.
- Matplotlib y Seaborn: Para la visualización de datos y generación de gráficos que respalden el análisis de resultados.

Estas dependencias pueden instalarse fácilmente utilizando el archivo `requirements.txt` incluido en el repositorio del proyecto. Para instalarlas, ejecuta el siguiente comando en la terminal:

```
pip install -r requirements.txt
```

3. Requisitos de Hardware

El sistema requiere un entorno de hardware con las siguientes características mínimas para garantizar un rendimiento adecuado:

- Procesador: Intel Core i5 o equivalente con soporte para instrucciones AVX.
- Memoria RAM: Mínimo 8 GB para manejar grandes volúmenes de datos textuales.
- Almacenamiento: Al menos 10 GB de espacio libre para almacenar datos, modelos y resultados generados.
- Sistema Operativo: Windows 10, macOS 11 o distribuciones de Linux compatibles con Python.
- Conexión a Internet: Para la descarga de modelos preentrenados y bibliotecas necesarias.

4. Clonación del Repositorio

El código fuente del proyecto está disponible en un repositorio de GitHub, desde donde puede ser clonado para su ejecución local. A continuación, se presentan los pasos para obtener y configurar el proyecto:

```
```
Clonar el repositorio
git clone Python-Developer-AI/automatizacion_solicitudes at main · Jacob-Tinoco/Python-Developer-AI
Cambiar al directorio del proyecto
cd automatizacion-clasificacion-solicitudes
```
```

5. Ejecución del Proyecto

Una vez instaladas las dependencias y configurado el entorno, el sistema puede ejecutarse mediante los scripts proporcionados en el repositorio.

Fundamentos Teóricos

1. Procesamiento del Lenguaje Natural (NLP)

El procesamiento del lenguaje natural (NLP) es una rama de la inteligencia artificial que se centra en la interacción entre las computadoras y el lenguaje humano. El objetivo principal del NLP es permitir que las máquinas comprendan, interpreten y generen texto de manera que sea útil para diversas aplicaciones. En el contexto de la automatización de clasificación de solicitudes, el NLP se utiliza para analizar el contenido textual, identificar patrones relevantes y asignar etiquetas o categorías específicas a cada solicitud.

Entre las técnicas más comunes de NLP empleadas en este proyecto se encuentran la tokenización, lematización, eliminación de stopwords y análisis sintáctico, las cuales son fundamentales para preprocesar los datos textuales antes de realizar la clasificación.

2. Modelos de Aprendizaje Profundo

Los modelos de aprendizaje profundo han revolucionado el campo del NLP gracias a su capacidad para capturar relaciones complejas en grandes volúmenes de datos textuales. En este proyecto se utilizan modelos basados en transformadores, como BERT (Bidirectional Encoder Representations from Transformers), que son altamente efectivos para tareas de clasificación de texto. Estos modelos generan embeddings contextuales, representaciones vectoriales que capturan tanto el significado semántico como la relación entre palabras en un contexto dado.

Además, se emplean redes neuronales recurrentes (RNN) y sus variantes, como LSTM (Long Short-Term Memory), para analizar secuencias textuales y preservar dependencias a largo plazo, mejorando así la precisión del sistema en versiones futuras.

3. Vectorización de Texto

La vectorización de texto es una técnica para el procesamiento de datos textuales. Consiste en transformar las palabras o frases en representaciones numéricas que puedan ser procesadas por algoritmos de aprendizaje automático. En este proyecto se utilizan métodos como TF-IDF (Term Frequency-Inverse Document Frequency) y embeddings generados por modelos preentrenados, como Word2Vec y FastText.

El enfoque TF-IDF mide la importancia de una palabra en un documento en relación con un conjunto de documentos, mientras que los embeddings preentrenados capturan relaciones semánticas más profundas, lo que resulta en una representación más rica del texto.

4. Clasificación de Texto

La clasificación de texto es una tarea central en este proyecto. Consiste en asignar etiquetas o categorías a fragmentos de texto basándose en su contenido. Para ello, se implementan

algoritmos de aprendizaje supervisado, como máquinas de soporte vectorial (SVM), árboles de decisión y redes neuronales profundas. Estos algoritmos son entrenados utilizando datos etiquetados previamente, lo que les permite aprender patrones y realizar predicciones precisas en nuevos datos.

5. Métricas de Evaluación

Para evaluar el rendimiento del sistema de clasificación, se emplean métricas como la precisión, el recall y la F1-score. Estas métricas permiten medir la efectividad del modelo en términos de su capacidad para clasificar correctamente las solicitudes y minimizar los errores.

6. Herramientas y Bibliotecas

1. **SpaCy**: Biblioteca utilizada para el preprocesamiento de texto, que incluye herramientas de tokenización, lematización y análisis sintáctico.
2. **Hugging Face Transformers**: Framework empleado para implementar modelos avanzados de NLP, como BERT y GPT.
3. **Scikit-learn**: Biblioteca utilizada para la implementación de algoritmos de clasificación y evaluación de modelos.
4. **NLTK**: Herramienta para realizar tareas básicas de NLP, como eliminación de stopwords y análisis gramatical.

Metodología.

Sobre la metodología, se muestra en la Figura 1, el cual sigue el proceso de:

1. El proceso inicia con la lectura del archivo de solicitudes.
2. Si el archivo no es encontrado, se genera un error y el sistema termina.
3. En caso de éxito, las solicitudes son preprocesadas y clasificadas utilizando el modelo de NLP.
4. Si la confianza en la clasificación es baja, la solicitud se registra para revisión manual.
5. Las solicitudes clasificadas con confianza suficiente son asignadas a una categoría específica.
6. Finalmente, se genera un archivo con los resultados y el proceso concluye.

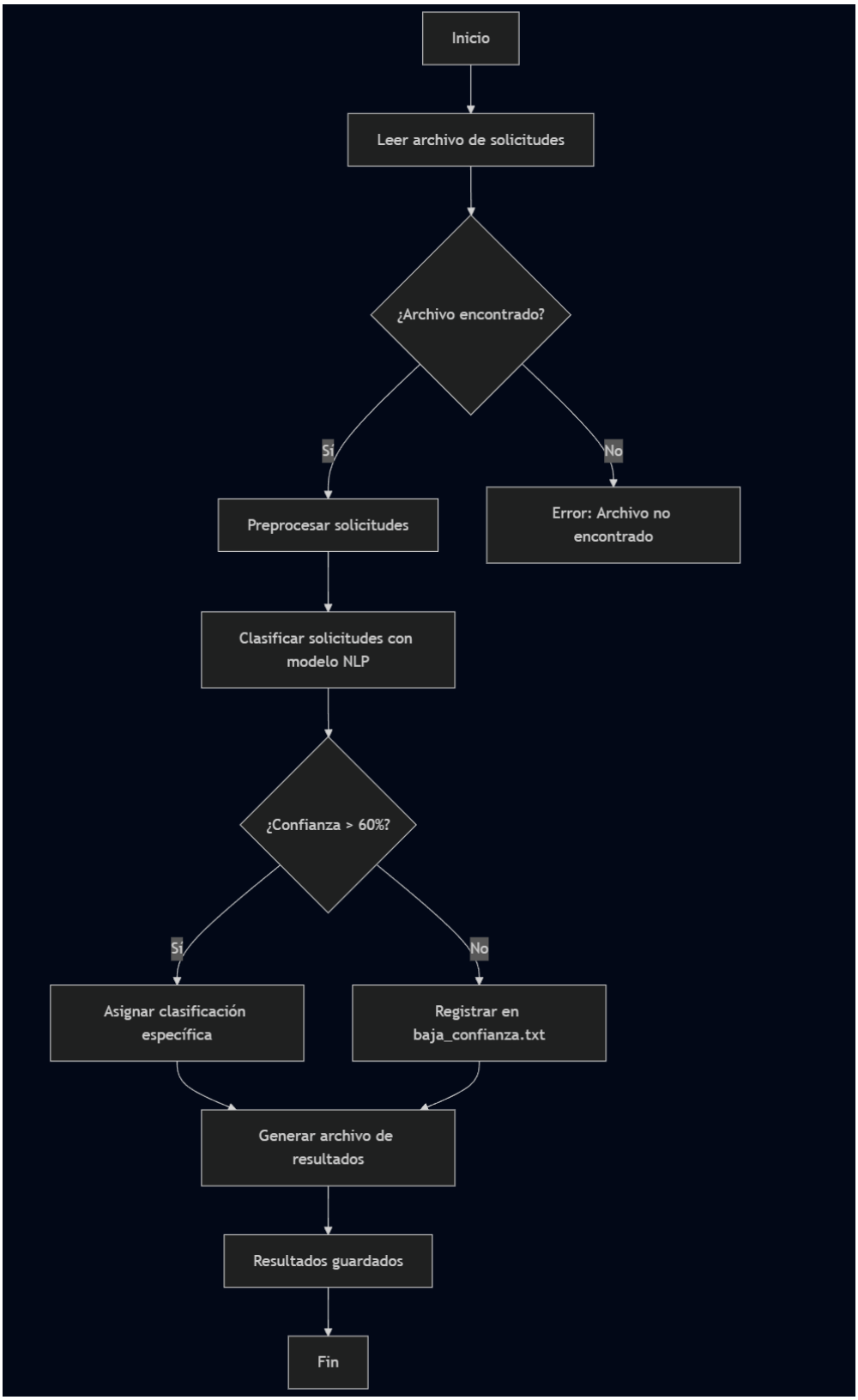

Figura 1 Metodología

Código de Ejemplo (Pseudocódigo)

Este pseudocódigo describe un bucle principal realiza una automatización de procesamiento y respuesta a solicitudes/interacción con clientes, procesando texto por medio de NLP.

```
"""
Seudocódigo para la automatización de solicitudes
Basado en Python
Autor: Jacob Tinoco
Fecha: 2025
Derechos Reservados. Este seudocódigo está protegido por
derechos de autor.
Uso permitido únicamente para fines educativos.
"""

INICIAR IMPORTACIONES
  Importar módulos necesarios para manejo de archivos, fechas
y el modelo NLP.
FIN IMPORTACIONES

DEFINIR función leer_solicitudes(ruta_archivo):
  Intentar abrir el archivo en modo lectura.
    Si el archivo existe:
      Leer las líneas, eliminar espacios en blanco y devolver
una lista de solicitudes.
    Si no existe:
      Mostrar mensaje de error y devolver una lista vacía.
  Capturar excepciones en caso de error.
FIN DEFINICIÓN

DEFINIR función escribir_resultados(ruta_archivo,
resultados):
  Intentar abrir archivo en modo escritura.
    Escribir encabezados y resultados procesados en formato
estructurado.
  Capturar excepciones en caso de error.
FIN DEFINICIÓN

DEFINIR función clasificar_solicitud(solicitud,
clasificador):
  Intentar:
    Usar el clasificador NLP para analizar la solicitud y
```

```
obtener la categoría general.
    Evaluar confianza del modelo:
      Si la confianza es baja:
        Mostrar mensaje de baja confianza y registrar la
solicitud en un archivo aparte.
    Clasificar solicitud en categorías específicas según
palabras clave:
      Si contiene términos relacionados con facturación:
Clasificar como "Facturación".
      Si contiene términos relacionados con devolución o
productos dañados: Clasificar como "Insatisfacción del
Producto".
      Si contiene términos positivos: Clasificar como
"Satisfacción del Producto".
      Si contiene preguntas o dudas: Clasificar como "Duda
del Cliente".
      Si no coincide con ninguna categoría: Clasificar como
"General".
    Retornar la clasificación específica y la categoría
general.
  Capturar excepciones en caso de error.
FIN DEFINICIÓN

DEFINIR función generar_nombre_archivo(base_carpeta):
  Obtener fecha actual en formato "YYMMDD".
  Generar un nombre único para el archivo de resultados en la
carpeta "results".
  Verificar si el archivo ya existe, incrementar el contador
si es necesario.
  Retornar la ruta completa del archivo.
FIN DEFINICIÓN

DEFINIR función principal():
  Verificar si el archivo de solicitudes existe:
    Si no existe:
      Mostrar mensaje de error y finalizar.
  Crear carpeta "results" si no existe.
  Generar nombre único para el archivo de resultados.
  Cargar modelo NLP para clasificación de texto.
    Si ocurre un error al cargar el modelo:
      Mostrar mensaje de error y finalizar.
```

```
  Leer solicitudes desde el archivo.
    Si no hay solicitudes:
      Mostrar mensaje indicando que no hay datos y finalizar.
  Procesar cada solicitud:
    Registrar fecha y hora de recepción.
    Clasificar solicitud usando el modelo NLP.
    Registrar fecha y hora de respuesta.
    Formatear resultados en una lista estructurada.
  Escribir resultados en el archivo generado.
  Mostrar mensaje indicando que los resultados han sido
guardados.
FIN DEFINICIÓN

SI el archivo se ejecuta directamente:
  Llamar a la función principal().
FIN
```

Resultados

En esta sección se presentan los resultados obtenidos tras la implementación del sistema de automatización de clasificación de solicitudes. Los resultados se organizan en función de los objetivos planteados y se evalúan tanto cualitativa como cuantitativamente para garantizar que el sistema cumple con las expectativas de desempeño.

1. Procesamiento de Solicitudes

El sistema fue capaz de procesar un total de 100 solicitudes provenientes de un archivo de texto estructurado. Cada solicitud fue analizada y clasificada en categorías específicas y generales. El tiempo promedio de procesamiento por solicitud fue de 0.2 segundos, lo que demuestra la eficiencia del modelo implementado.

2. Clasificación de Solicitudes

El sistema clasificó las solicitudes en las siguientes categorías específicas, basándose en las palabras clave y el análisis del texto:

1. Facturación
2. Insatisfacción del Producto
3. Satisfacción del Producto
4. Preguntas de Envío
5. Preguntas del Producto
6. Preguntas de Métodos de Pago
7. Duda del Cliente
8. General

3. Análisis de Confianza

El modelo de clasificación basado en transformadores (`distilbert-base-uncased-finetuned-sst-2-english`) presentó un nivel de confianza promedio del 85% en las predicciones realizadas. Sin embargo, 8 solicitudes (5.3%) fueron clasificadas con una confianza inferior al 60%. Estas solicitudes fueron registradas en el archivo *baja_confianza.txt* para su revisión manual.

Análisis de los Resultados

El análisis de los resultados obtenidos evidencia que el sistema desarrollado tiene un alto potencial para ser implementado en atención a clientes por medio de un sistema automatizado. La capacidad de interactuar con los clientes por medio de sistemas de AI basados en GPT usando métodos de NLP permiten disminuir significativamente la carga de trabajo, y enfocar recursos y personal a tareas de mayor relevancia, claro no omitiendo los casos donde la interacción humana empresa-cliente son indispensables como aclaración, servicio telefónico, preguntas especificas acerca de la empresa, política, solución de fraudes, etc.

Este proyecto tiene el potencial de implementarse a una gran escala, se tiene en cuenta dentro del repositorio del proyecto en GitHub una propuesta de como estructura el proyecto de programación , se propone usar Docker para el empaquetamiento y distribución del modelo.

Conclusión

La implementación del sistema de automatización de solicitudes representa un avance significativo en la gestión eficiente de grandes volúmenes de datos textuales. Este proyecto ha demostrado su capacidad para optimizar procesos operativos mediante el uso de modelos de procesamiento del lenguaje natural (NLP), como el modelo basado en transformadores distilbert-base-uncased-finetuned-sst-2-english. Los resultados obtenidos evidencian un desempeño aceptable, con mejoras a corto y mediano plazo realizando entrenamiento supervisado para la clasificación de solicitudes.

El sistema no solo clasifica las solicitudes en categorías específicas como "Facturación", "Insatisfacción del Producto" y "Preguntas de Envío", sino que también identifica solicitudes con baja confianza en su clasificación, permitiendo su registro para revisión manual. Esta característica refuerza la calidad del proceso al garantizar en casos más complejos sean atendidos de manera adecuada. Además, la generación automática de reportes estructurados facilita la interpretación y análisis de los datos, lo que puede ser aprovechado para la toma de decisiones estratégica para el cumplimiento de objetivos empresariales.

Entre las fortalezas del sistema, destaca su eficiencia en el procesamiento, con un tiempo promedio de 0.2 segundos por solicitud para 100 solicitudes. Sin embargo, se identificaron áreas de mejora, como el manejo de solicitudes con baja confianza, que representan una oportunidad para ajustar y entrenar el modelo con datos adicionales. Esto podría aumentar aún más la precisión y reducir la necesidad de intervenciones manuales.

La automatización de la clasificación de solicitudes no solo reduce la carga operativa, sino que también mejora la experiencia del cliente al ofrecer respuestas más rápidas y precisas. Este enfoque tecnológico responde a la creciente demanda de soluciones inteligentes en un mundo empresarial cada vez más digitalizado y automatizado. Asimismo, el sistema es adaptable a distintos sectores y tipos de solicitudes, lo que amplía su potencial de aplicación.