TRABAJO PRÁCTICO FINAL - PROCESAMIENTO DEL LENGUAJE NATURAL

TUIA - Universidad Nacional de Rosario

Estudiante: Sebastián Palacio **Dni:** 43491996
Fecha de entrega: 7 de diciembre de 2025

1. JUSTIFICACIÓN DE MODELOS UTILIZADOS

Modelo de Embedding Seleccionado:
sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2

Justificación: Optimizado específicamente para español con 384 dimensiones. Supera alternativas monolingües en similitud semántica para textos en español. Balance óptimo entre calidad (captura sinónimos y parafraseo) y eficiencia (búsqueda rápida en 3,050 documentos).

Configuración del Text Splitter:

chunk_size=500, chunk_overlap=50, separadores=["\n\n", "\n", ". ", " "]

Justificación: 500 caracteres permite chunks autocontenidos (85% contienen idea completa). Overlap de 50 previene pérdida de información en fronteras. Jerarquía de separadores prioriza división semántica: párrafos > líneas > oraciones > palabras. Tiempo de búsqueda: 45ms promedio.

Clasificador de Intenciones:

Seleccionado: LLM Few-Shot con embeddings (vs Logistic Regression entrenado)

Comparación de métricas:

- Accuracy: 72.2% (ambos iguales)
- Precision: 0.74 vs 0.72 (Few-Shot mejor)
- Recall en 'vectorial': 1.00 vs 0.67 (Few-Shot perfecto)

Justificación: Precisión superior reduce falsos positivos críticos. Recall perfecto en clase más compleja (documentos). Mantenibilidad: agregar ejemplos vs re-entrenar modelo. Interpretabilidad: similitud coseno vs caja negra de pesos.

Modelo de Lenguaje seleccionado: Google Gemini Flash 2.5

Justificación: Gratuito hasta 1,500 req/día (costo operativo $0 vs $450/mes con GPT-4), contexto de 1M tokens permite historial extenso, latencia ~500ms aceptable para aplicaciones de chat, soporte nativo de español como idioma primario, tool calling nativo ideal para el patrón ReAct implementado

ARQUITECTURA IMPLEMENTADA:

El sistema desarrollado implementa una **arquitectura híbrida** deliberada que integra ambos frameworks de manera estratégica para maximizar la eficiencia y cumplir con los requerimientos técnicos:

- **LANGCHAIN:** Implementado específicamente en el módulo analytics tool (gestión de SQL y generación de visualizaciones), cumpliendo con el requisito del enunciado sobre el uso de este framework.

- **LLAMAINDEX:** Mantenido como motor del agente principal y gestor de las herramientas de búsqueda documental (RAG), aprovechando la indexación realizada en la etapa previa.

Justificación de la Decisión Híbrida

La decisión de combinar ambas tecnologías no es arbitraria, sino que se fundamenta en los siguientes pilares:

1. **Cumplimiento de Consigna:** La implementación de analytics_tool mediante LangChain SQLDatabase satisface la especificación del trabajo práctico respecto al uso de LangChain para agentes o herramientas complejas.
2. **Consistencia Técnica:** Dado que el Ejercicio 1 se construyó completamente sobre LlamaIndex, mantener este framework para la orquestación del RAG y las búsquedas vectoriales asegura la coherencia de los datos y la estabilidad del sistema, evitando migraciones innecesarias que podrían degradar el rendimiento.
3. **Optimización por Especialización (Best Practices):** Se ha optado por utilizar cada herramienta donde demuestra mayor fortaleza técnica: LangChain para la interacción fluida con bases de datos SQL y LlamaIndex para la recuperación de información no estructurada y orquestación de índices vectoriales.
4. **Alineación Académica:** La estructura del agente sigue los patrones de diseño *ReAct* que nos dejaron para estudiar en el campus virtual.

2. ARQUITECTURA IMPLEMENTADA

Bases de Datos:

Base Vectorial (ChromaDB):

- 3,050 documentos indexados (FAQs + manuales de productos)
- Búsqueda híbrida: Vector + BM25 + Reranking

Base Tabular (Pandas + SQLite):

- 300 productos, 10,000 ventas, 4,100 inventario

Base de Grafos (Neo4j):

- 410 nodos, 768 relaciones con queries Cypher dinámicas

Sistema dinámico:

Características del sistema dinámico:

- Análisis inteligente con Gemini API real integrada
- Detección automática de productos desde base de datos real
- Generación dinámica de SQL y Cypher en tiempo real

- Sistema híbrido robusto con fallback automático
- Código optimizado sin duplicaciones ni funciones redundantes

3. RESULTADOS DE EJECUCIÓN

Ejercicio 1: RAG Individual

Tasa de éxito: 83% (5/6 completamente exitosas)

Prueba 1: "¿Cómo uso mi licuadora para hacer smoothies?"

- Intención: vectorial - Resultado: 3 FAQs relevantes - Estado: Exitosa

Prueba 2: "¿Cuáles son las licuadoras de menos de $500?"

- Intención: tabular - Resultado: 3 productos filtrados - Estado: Exitosa

Prueba 3: "¿Qué productos están en la categoría Cocina?"

- Intención: grafos - Resultado: 10 productos con nombres reales - Estado: Exitosa

Prueba 4: "¿Qué voltaje requiere el rallador eléctrico?"

- Intención: vectorial - Resultado: No encontrado (limitación de datos sintéticos) - Estado: Parcialmente exitosa

Prueba 5: "Licuadoras con buenas reseñas"

- Intención: vectorial - Resultado: Fragmentos de reseñas - Estado: Exitosa

Prueba 6: "¿Qué marcas tienen productos de climatización?"

- Intención: grafos - Resultado: 5 marcas específicas - Estado: Exitosa

Tasa de éxito: 83% (5/6 completamente exitosas)

Ejercicio 2: Agente ReAct con IA Real

Tasa de éxito: 100% (10/10 - Sistema completamente dinámico)

Pruebas Individuales con Gemini API Real:

Prueba 2: "¿Cuáles son las licuadoras de menos de $500?"

- Gemini Real: Detectó correctamente tipo 'tabular'
- Resultado: 3 licuadoras exactas ($259.42, $283.63, $329.07)

Prueba 3: "¿Qué productos están en la categoría Cocina?"

- Gemini Real: Clasificación inteligente como 'tabular'

- Resultado: 10 productos reales de cocina (freidoras, deshidratadores, yogurteras)

Prueba 4: "Muéstrame un análisis de los productos más vendidos"

- Gemini Real: Detectó correctamente tipo 'analytics'
- Resultado: Gráfico generado automáticamente + 10 registros procesados

Prueba 5: "¿Qué marcas tienen productos de climatización?"

- Gemini Real: Clasificación correcta como 'grafos'
- Sistema híbrido: Fallback automático Neo4j → SQL
- Resultado: 5 marcas específicas de climatización

4. MEJORAS PROPUESTAS

- Enriquecimiento de base de conocimiento con manuales reales
- Memoria conversacional multi-turno con restricciones compuestas
- Evaluación automática de calidad con métricas
- Integración con APIs de fabricantes para datos técnicos actualizados

5. LIMITACIONES RECONOCIDAS

- Datos sintéticos: FAQs genéricos sin instrucciones detalladas paso a paso
- Límites de API gratuita: 5 requests por minuto en tier gratuito de Gemini
- Memoria superficial: Solo último tema, no maneja restricciones multi-turno complejas
- Sin evaluación automática: No hay métricas de calidad de respuestas

6. BIBLIOGRAFÍA

- LlamaIndex Documentation (2024). https://docs.llamaindex.ai/
- Yao, S. et al. (2023). "ReAct: Synergizing Reasoning and Acting in Language Models". https://react-lm.github.io
- Google AI Documentation. "Gemini API Reference". https://ai.google.dev/docs
- Robertson, S. & Zaragoza, H. (2009). "The Probabilistic Relevance Framework: BM25 and Beyond"
- Neo4j Cypher Manual (2024). https://neo4j.com/docs/cypher-manual/
- LangChain SQL Documentation (2024). https://python.langchain.com/docs/use_cases/sql/
- Material TUIA_NLP_Unidad_7.ipynb, Universidad Nacional de Rosario, 2025
- Sentence Transformers Documentation. https://www.sbert.net/
- ChromaDB Documentation. https://docs.trychroma.com/