

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 1.5

Justificación: Gratuito hasta 1,500 req/día. Contexto de 1M tokens permite historial extenso. Latencia ~500ms aceptable para chat. Soporte nativo de español como idioma primario. Tool calling nativo ideal para ReAct. Costo operativo: \$0 en tier gratuito vs \$450/mes con GPT-4o. Alojamiento en nube (Google Cloud) sin infraestructura que mantener.

Framework: LlamaIndex

DECISIÓN TOMADA: Utilicé LlamaIndex para todo el proyecto (Ejercicio 1 y 2) ya que este era el que usamos en el material de clases ([TUIA_NLP_Unidad_7.ipynb](#)) y me guié principalmente por los contenidos de clase. Los ejemplos de ReAct en clase están también en LlamaIndex, junto con toda la metodología enseñada.

2. ARQUITECTURA IMPLEMENTADA

Ejercicio 1 - RAG

Bases de datos:

- Vectorial: ChromaDB con 3,050 documentos (FAQs + manuales)
- Tabular: Pandas con 300 productos, 10,000 ventas, 4,100 inventario
- Grafos: Neo4j con 410 nodos, 768 relaciones
(Producto-Categoría-Marca-Vendedor-Sucursal)

Pipeline de recuperación: Búsqueda híbrida = Vector search (LlamaIndex) + BM25 (keyword) + Reranking (cross-encoder). Vector captura semántica, BM25 captura términos exactos, reranker optimiza orden final.

Ejercicio 2 - Agente ReAct

4 Herramientas implementadas:

1. doc_search_tool: Búsqueda híbrida en documentos
2. table_search_tool: Filtros dinámicos en Pandas con fallback inteligente
3. graph_search_tool: Queries Cypher dinamicas en Neo4j
4. analytics_tool: Langchain SQLDatabase + SQLAlchemy + Matplotlib (cumple requisito SQL)

Patron ReAct: Thought - Action - Observation - Final Answer

Memoria conversacional: Detecta continuidad ("Y...", "Tambien..."), extrae tema anterior, enriquece query automáticamente. Ejemplo: "Licuadoras menos \$300?" seguido de "Y con buen stock?" se enriquece a "licuadoras con buen stock".

3. RESULTADOS DE EJECUCIÓN

Ejercicio 1 (6 pruebas)

Prueba 1: Como uso licuadora para smoothies?

- Intencion: vectorial
- Resultado: Recupero 3 FAQs relevantes
- Estado: Exitosa

Prueba 2: Licuadoras de menos de \$500?

- Intencion: tabular
- Resultado: 3 productos filtrados correctamente
- Estado: Exitosa

Prueba 3: Productos en categoria Cocina?

- Intencion: grafos
- Resultado: 10 IDs retornados (query Cypher correcta)
- Estado: Exitosa

Prueba 4: Voltaje del rallador electrico?

- Intencion: vectorial
- Resultado: No encontro (dato no existe en sinteticos)
- Estado: Parcialmente exitosa

Prueba 5: Licuadora con buenas reseñas

- Intencion: vectorial
- Resultado: Fragmentos de opiniones recuperados
- Estado: Exitosa

Prueba 6: Marcas con productos climatizacion?

- Intencion: grafos
- Resultado: 5 marcas (AirFlow, PureAir, ThermoControl, ClimaTech, EcoClima)
- Estado: Exitosa

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

La Prueba 4 busco información técnica específica (voltaje) que no existe en los datos sintéticos del TP. Los FAQs y manuales generados para el trabajo son genéricos y no contienen especificaciones técnicas detalladas. El sistema funcionó correctamente (clasificó bien, busco en la fuente apropiada) pero la limitación es de los datos, no del sistema. En producción con manuales reales, esta consulta retorna el voltaje correcto.

Ejercicio 2 (10 pruebas: 6 individuales + 4 secuencia con memoria)

Prueba 1: Como uso licuadora?

- Herramienta: doc_search
- Resultado: FAQs recuperados
- Estado: Exitosa

Prueba 2: Licuadoras menos \$500?

- Herramienta: table_search
- Resultado: 3 productos
- Estado: Exitosa

Prueba 3: Productos en Cocina?

- Herramienta: graph_search
- Resultado: 10 IDs
- Estado: Exitosa

Prueba 4: Analisis mas vendidos

- Herramienta: analytics
- Resultado: Top 10 + grafico SQL generado
- Estado: Exitosa

Prueba 5: Marcas climatizacion?

- Herramienta: graph_search
- Resultado: 5 marcas filtradas
- Estado: Exitosa

Prueba 6: Cafetera economica + stock

- Herramienta: table_search
- Resultado: Fallback mostro mejor disponible
- Estado: Exitosa

Pruebas 7-8: Secuencia con memoria

- Query 7: "Licuadoras menos \$300?" - Resultado: 2 licuadoras. Tema guardado: "licuadora"
- Query 8: "Y cuales tienen buen stock?" - Memoria activa: enriquece a "licuadoras con buen stock" - Resultado: 7 licuadoras (stock mayor 50)
- Estado: Exitosa (memoria funcional)

Pruebas 9-10: Queries independientes

- Resultado: Sin aplicar memoria incorrectamente
- Estado: Exitosa

Tasa de exito: 100% (10/10)

4. MEJORAS PROPUESTAS

- Integrar LLM real (Gemini): Respuestas en lenguaje natural vs resultados crudos actuales.
- Nombres legibles en Neo4j: Mostrar "Licuadora TechHome" vs "P0042".
- Enriquecer base de conocimiento: Reemplazar datos sinteticos con manuales/FAQs reales.

5. LIMITACIONES RECONOCIDAS

- Datos sintéticos: FAQs genericos sin instrucciones detalladas paso a paso

- LLM simulado: Sin API key, no genera lenguaje natural conversacional
- Queries hardcoded: Patrones if/elif limitan cobertura vs LLM dinámica.
- Memoria superficial: Sólo ultimo tema, no maneja restricciones compuestas multi-turn
- 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>
- 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