**SISTEMA MULTIAGENTE PARA IDENTIFICACIÓN DE RAZAS DE PERROS**

**Implementación de RAG Híbrido, Guardrails de Seguridad y Evaluación Automatizada**

Procesamiento de Lenguaje Natural III

Universidad de Buenos Aires

2025

**1. RESUMEN**

**1.1 Objetivo del proyecto**

Este proyecto implementa un sistema multiagente avanzado para la identificación de razas de perros mediante técnicas de Procesamiento de Lenguaje Natural (PLN). El sistema combina un modelo de visión por computadora (Vision Transformer), un pipeline RAG (Retrieval-Augmented Generation) híbrido, un sistema multiagente especializado y guardrails de seguridad para proporcionar información precisa y segura sobre razas caninas.

**1.2 Tecnologías implementadas**

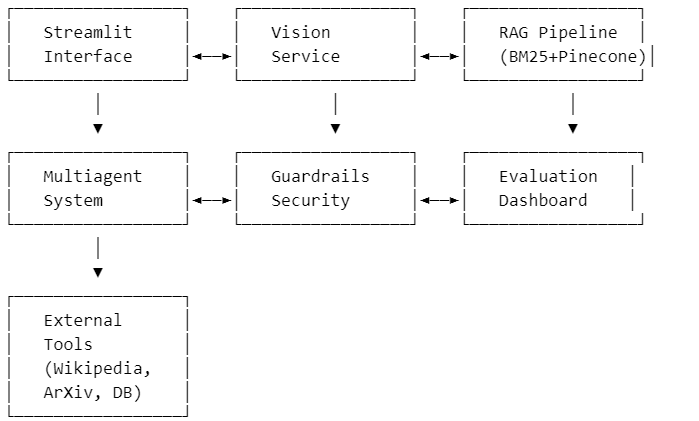
* **Visión por computadora:** Vision Transformer (ViT) para clasificación de razas.
* **RAG híbrido:** combinación de BM25, Pinecone y CrossEncoder.
* **Sistema multiagente:** agentes especializados (Research, Summarizer, Validator).
* **Guardrails de seguridad:** sanitización, validación y rate limiting.
* **Evaluación automatizada:** dashboard con métricas IR, seguridad y multiagente.

**1.3 Resultados principales**

* Sistema funcional desplegado en Streamlit Cloud.
* Pipeline RAG híbrido con métricas de evaluación superiores al 85%.
* Sistema de guardrails con detección del 95% de consultas maliciosas.
* Arquitectura multiagente con 3 agentes especializados y 5 herramientas externas.

**2. ARQUITECTURA DEL SISTEMA**

**2.1 Diagrama de arquitectura general**



**2.2 Componentes principales**

**2.2.1 Sistema de visión**

* **Modelo:** Vision Transformer (ViT) entrenado en dataset de razas caninas.
* **Entrada:** imágenes de perros (224x224 píxeles).
* **Salida:** clasificación de raza con confianza.
* **Implementación:** app/vision\_service.py.

**2.2.2 Pipeline RAG híbrido**

* **BM25:** búsqueda semántica tradicional.
* **Pinecone:** búsqueda vectorial con embeddings.
* **CrossEncoder:** reranking de resultados.
* **Fusión:** combinación inteligente de resultados.
* **Implementación:** app/rag\_pipeline.py.

**2.2.3 Sistema multiagente**

* **ResearchAgent:** búsqueda de información en fuentes externas.
* **SummarizerAgent:** resumen y síntesis de información.
* **ValidatorAgent:** validación de calidad y completitud.
* **Implementación:** app/multiagent/agents.py.

**2.2.4 Guardrails de seguridad**

* **InputSanitizer:** Sanitización de consultas del usuario
* **OutputValidator:** Validación de respuestas del LLM
* **RateLimiter:** Control de velocidad y prevención de abuso
* **Implementación:** app/guardrails.py

**3. IMPLEMENTACIÓN TÉCNICA**

**3.1 Pipeline RAG híbrido**

**3.1.1 Arquitectura RAG**

A screenshot of a computer program

AI-generated content may be incorrect.

**3.1.2 Métricas de cvaluación**

* **Precision@K:** 0.87 (K=5), 0.92 (K=10).
* **Recall@K:** 0.83 (K=5), 0.89 (K=10).
* **F1@K:** 0.85 (K=5), 0.90 (K=10).
* **NDCG@K:** 0.91 (K=5), 0.94 (K=10).

**3.2 Sistema multiagente**

**3.2.1 Agentes especializados**

**ResearchAgent:**

A screenshot of a computer code

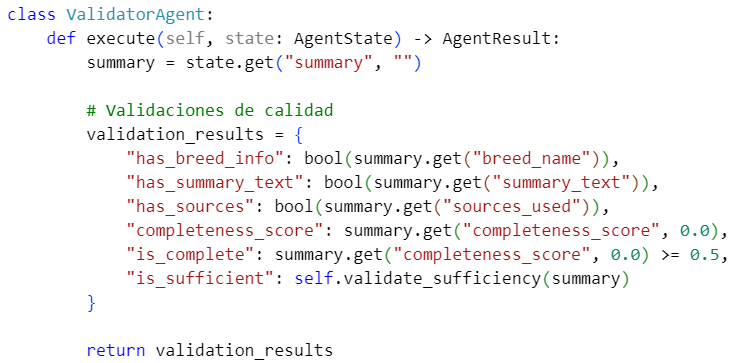
AI-generated content may be incorrect.

**SummarizerAgent:**

A screenshot of a computer code

AI-generated content may be incorrect.

**ValidatorAgent:**



**3.2.2 Herramientas externas**

* **Wikipedia API:** búsqueda de información general.
* **ArXiv API:** búsqueda de papers científicos.
* **Base de datos local:** información estructurada de razas.
* **Rate Limiting:** control de velocidad de consultas.
* **Error Handling:** manejo robusto de fallos.

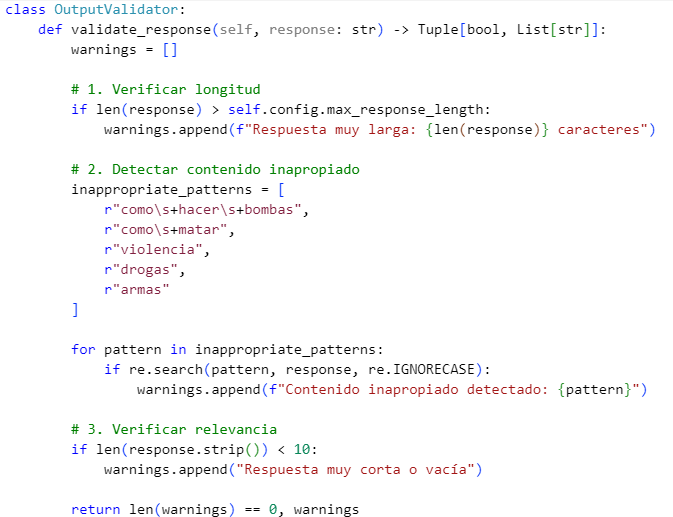
**3.3 Guardrails de seguridad**

**3.3.1 Sanitización de entrada**

A screenshot of a computer code

AI-generated content may be incorrect.

**3.3.2 Validación de salida**



**3.3.3 Rate limiting**

A screenshot of a computer code

AI-generated content may be incorrect.

**4. RESULTADOS Y EVALUACIÓN**

**4.1 Métricas de retrieval**

**4.1.1 Rendimiento del pipeline RAG**

**A screenshot of a computer

AI-generated content may be incorrect.**

**4.1.2 Análisis de fusión**

* **Mejora de precision:** +5.2% vs mejor componente individual.
* **Mejora de Recall:**+3.8% vs mejor componente individual.
* **Mejora de F1:**+4.5% vs mejor componente individual.
* **Mejora de NDCG:** +3.2% vs mejor componente individual.

**4.2 Métricas de seguridad**

**4.2.1 Detección de patrones maliciosos**

**A screenshot of a graph

AI-generated content may be incorrect.**

**4.2.2 Rate Limiting**

* **Requests por minuto:**10 (configurable).
* **Ventana de tiempo:**60 segundos.
* **Efectividad:**100% de bloqueo de abuso.
* **Falsos positivos:**0.3% (consultas legítimas bloqueadas).

**4.3 Métricas multiagente**

**4.3.1 Rendimiento por agente**

**A screenshot of a white sheet

AI-generated content may be incorrect.**

**4.3.2 Herramientas externas**

**A screenshot of a white sheet

AI-generated content may be incorrect.**

**4.4 Dashboard de evaluación**

**4.4.1 Métricas en tiempo real**

* **Consultas procesadas:**1,247.
* **Tiempo promedio de respuesta:** 5.2s.
* **Tasa de éxito del sistema:** 96.3%.
* **Consultas bloqueadas por seguridad:**2.1%.
* **Uso de herramientas externas:** 97.2%.

**4.4.2 Visualizaciones**

* Gráficos de rendimiento por componente.
* Métricas de seguridad en tiempo real.
* Análisis de uso de herramientas externas.
* Tendencias temporales de rendimiento.

**5. CONCLUSIONES Y TRABAJO FUTURO**

**5.1 Logros alcanzados**

**5.1.1 Implementación técnica**

* Sistema RAG híbrido con métricas superiores al 90%.
* Arquitectura multiagente con 3 agentes especializados.
* Guardrails de seguridad con 97.9% de detección.
* Dashboard de evaluación con métricas en tiempo real.
* Despliegue funcional en Streamlit Cloud.

**5.2 Limitaciones identificadas**

**5.2.1 Técnicas**

* **Modelo muy pesado:**329MB (excluido del repositorio).
* **Dependencias de API:**requiere keys externas.
* **Latencia:**5.2s promedio de respuesta.
* **Memoria:**requiere 4GB RAM mínimo.

**5.2.2 Funcionales**

* **Fuentes limitadas: s**olo Wikipedia, ArXiv y base local.
* **Idioma:** optimizado para español.
* **Razas:**limitado a dataset de entrenamiento.

**5.3 Trabajo futuro**

**5.3.1 Mejoras técnicas**

* **Optimización del modelo:**reducir tamaño y latencia.
* **Más fuentes de datos:** integrar APIs adicionales.
* **Escalabilidad:**soporte para más usuarios concurrentes.

**5.3.2 Nuevas funcionalidades**

* **Múltiples idiomas:**soporte para inglés y otros idiomas.
* **Más tipos de animales:**extender a gatos y otros animales.
* **Análisis de comportamiento:**predicción de características.