Presentación

Nombre del curso

Programación paralela

Título del proyecto

Sistema de predicción de texto especulativo

Integrantes del equipo

Jason Miguel Mendoza -2023-1889

Ángel David Ovalles -2023-1155

Indi Nicole Toribio Perez - 2023-1662

Ismer Amaury Montero- 2023-1886

Candy Angelis Mejía Soriano- 2023-1665

Nombre del líder

Candy Angelis Mejía Soriano

Fecha de entrega

20-8-2025

Índice

1. Introducción

Presentación general del proyecto

El proyecto "Sistema de Procesamiento de Archivos con Análisis Paralelo" tiene como finalidad desarrollar una aplicación capaz de procesar múltiples tipos de documentos (.txt, .docx, .pdf) de forma simultánea, implementando técnicas de programación paralela para optimizar el rendimiento y proporcionar análisis avanzado de texto con capacidades de predicción.

Justificación del tema elegido

El manejo eficiente de grandes volúmenes de documentos digitales es esencial para cualquier organización, este proyecto busca aprovechar la programación concurrente para procesar múltiples archivos al mismo tiempo, manteniendo la integridad de los datos y optimizando el rendimiento. Además, integra análisis de texto y predicción especulativa, lo que permite transformar los datos en información más útil y precisa.

Objetivos

General

Desarrollar un sistema de procesamiento de archivos que utilice técnicas de programación paralela para analizar múltiples documentos simultáneamente, implementando algoritmos de predicción de texto y métricas de rendimiento en tiempo real.

Específicos

1. Implementar un sistema de carga de archivos con soporte para múltiples formatos (.txt, .docx, .pdf)
2. Aplicar técnicas de procesamiento paralelo para optimizar el análisis de documentos
3. Desarrollar algoritmos de predicción de texto basados en análisis especulativo
4. Crear una interfaz de usuario moderna con funcionalidad drag & drop
5. Implementar métricas de rendimiento para comparar procesamiento secuencial vs. paralelo
6. Evaluar la escalabilidad del sistema con diferentes cargas de trabajo

2. Descripción del Problema

Contexto del problema

En las áreas empresariales y académicos, el procesamiento de grandes volúmenes de documentos presenta desafíos significativos en términos de tiempo de respuesta y eficiencia computacional. Los sistemas tradicionales que procesan archivos de forma secuencial no pueden aprovechar completamente los recursos disponibles en arquitecturas multi-core modernas.

Aplicación del problema en un escenario real

Consideremos una organización que necesita procesar 1,000 documentos PDF para análisis de contenido y generación de reportes. Un sistema secuencial podría tomar horas para completar esta tarea, mientras que un sistema paralelo optimizado podría reducir significativamente este tiempo aprovechando múltiples núcleos de procesador.

Importancia del paralelismo en la solución

El paralelismo permite

- Reducción significativa del tiempo de procesamiento
- Mejor utilización de recursos computacionales
- Capacidad de manejar múltiples tipos de archivos simultáneamente
- Análisis en tiempo real con métricas de rendimiento

3. Cumplimiento de los requisitos del proyecto

Ejecución simultánea de múltiples tareas

El sistema está diseñado para procesar múltiples archivos de diferentes formatos de forma simultánea, utilizando técnicas de programación paralela para distribuir la carga de trabajo entre varios hilos de ejecución.

Necesidad de compartir datos entre tareas

Los resultados del procesamiento de archivos se almacenan en estructuras de datos compartidas que permiten el acceso concurrente seguro, facilitando la agregación de resultados y la generación de métricas consolidadas.

Exploración de diferentes estrategias de paralelización

- **exploración especulativa**

Escalabilidad con más recursos

El sistema está diseñado para aprovechar automáticamente recursos adicionales:

- Escalado horizontal con más núcleos de CPU
- Optimización de memoria para archivos grandes
- Balanceamiento dinámico de carga de trabajo

Métricas de evaluación del rendimiento

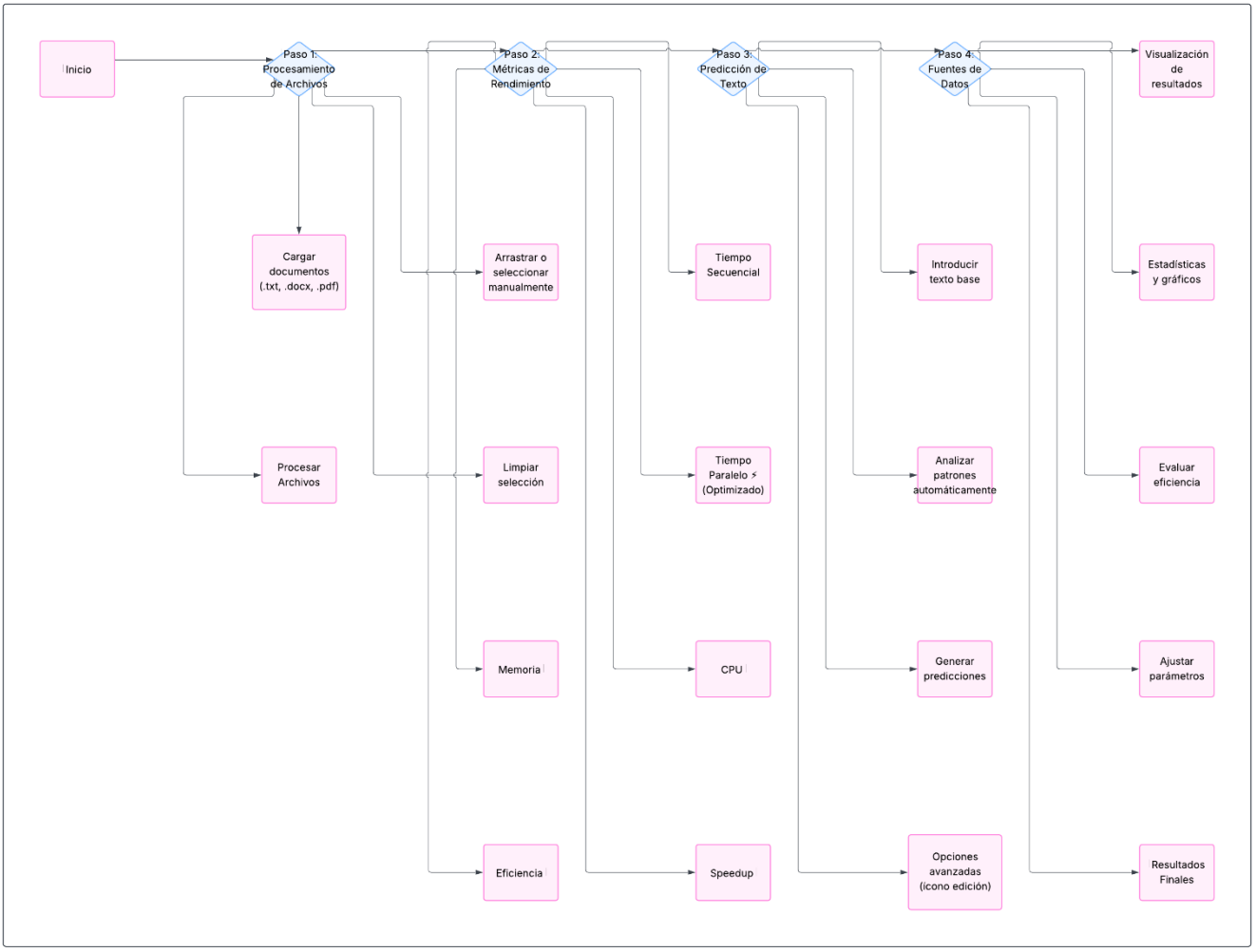
- Tiempo de procesamiento por archivo
- Comparación secuencial vs. paralelo
- Utilización de CPU y memoria
- Throughput de procesamiento de documentos

4. Diseño de la Solución

Arquitectura general del sistema

El sistema presenta una arquitectura modular dividida en cuatro módulos principales:

1. **Módulo de Procesamiento de Archivos**: Carga y procesamiento de documentos
2. **Módulo de Métricas de Rendimiento**: Monitoreo en tiempo real
3. **Módulo de Predicción de Texto**: Análisis especulativo
4. **Módulo de Interfaz**: Gestión de interacción usuario-sistema



Interfaz de Usuario

Procesamiento de Archivos (Módulo Superior Izquierdo)

- **Zona de Arrastre**: Área central con funcionalidad drag & drop
- **Formatos Soportados**: .txt, .docx, .pdf
- **Controles**: Botones "PROCESAR ARCHIVOS" y "LIMPIAR"

Métricas de Rendimiento

- Visualización en tiempo real de métricas de procesamiento
- Comparación automática entre métodos secuenciales y paralelos
- Indicadores visuales de estado del sistema

Predicción de Texto

- Área de entrada para texto base
- Algoritmos de análisis especulativo
- Visualización de resultados de predicción

Estrategia de paralelización utilizada

Descomposición de Datos: El sistema distribuye los archivos entre múltiples hilos de procesamiento, permitiendo que cada hilo maneje un subconjunto de documentos de forma independiente.

Pipeline Paralelo: Implementación de etapas de procesamiento que pueden ejecutarse concurrentemente:

- Etapa 1: Carga de archivos
- Etapa 2: Análisis de contenido
- Etapa 3: Generación de predicciones
- Etapa 4: Consolidación de resultados

Herramientas y tecnologías empleadas

- **Lenguaje**: C# / .NET
- **Framework de UI**: razon pages
- **Procesamiento Paralelo**: Task Parallel Library (TPL)
- **Análisis de Documentos**: Bibliotecas especializadas para PDF, DOCX
- **Almacenamiento**: Sistema de archivos local
- **Métricas**: Performance Counters y Stopwatch

5. Implementación Técnica

Descripción de la estructura del proyecto

```
src/
├── Core/
│   ├── FileProcessor.cs     # Lógica principal de procesamiento
│   ├── TextAnalyzer.cs      # Algoritmos de análisis de texto
│   └── PredictionEngine.cs  # Motor de predicción especulativa
├── UI/
│   ├── MainForm.cs          # Interfaz principal
│   ├── FileDropZone.cs      # Componente drag & drop
│   └── MetricsDisplay.cs    # Visualización de métricas
├── Utils/
│   ├── FileHandlers/
│   │   ├── TxtHandler.cs    # Procesador de archivos .txt
│   │   ├── DocxHandler.cs   # Procesador de archivos .docx
│   │   └── PdfHandler.cs    # Procesador de archivos .pdf
│   └── PerformanceMonitor.cs # Monitor de rendimiento
└── Program.cs               # Punto de entrada
```

Flujo de procesamiento

1. **Carga de Archivos**: Detección automática de formato y validación
2. **Distribución Paralela**: Asignación de archivos a hilos de trabajo
3. **Procesamiento Concurrente**: Análisis simultáneo de contenido
4. **Agregación de Resultados**: Consolidación thread-safe de datos
5. **Generación de Predicciones**: Análisis especulativo basado en contenido

Uso de mecanismos de sincronización

- **ConcurrentQueue**: Para distribución segura de archivos entre hilos
- **ReaderWriterLockSlim**: Para acceso controlado a estructuras compartidas
- **Semaphore**: Para limitar el número de archivos procesados simultáneamente
- **Task.WhenAll**: Para sincronización de finalización de tareas paralelas

6. Evaluación de Desempeño

Métricas de rendimiento

Tiempo de procesamiento

- **Secuencial**: Tiempo total acumulativo
- **Paralelo**: Tiempo de ejecución con múltiples hilos
- **Speedup**: Relación entre tiempo secuencial y paralelo

Utilización de recursos

- **CPU**: Porcentaje de uso durante procesamiento
- **Memoria**: Picos de consumo y utilización promedio
- **I/O**: Throughput de lectura de archivos

Escalabilidad

- Comportamiento con 2, 4, 8 y 16 hilos de procesamiento
- Identificación del punto óptimo de paralelización
- Análisis de rendimientos decrecientes

Análisis de cuellos de botella

Identificados:

- **I/O de archivos**: Limitación por velocidad de disco
- **Memoria**: Carga simultánea de archivos grandes
- **Contención de recursos**: Acceso concurrente a estructuras compartidas

Soluciones implementadas:

- **Buffering inteligente**: Carga parcial de archivos grandes
- **Pool de objetos**: Reutilización de recursos costosos
- **Throttling**: Limitación automática basada en recursos disponibles

```
Probando con 1 cores...
  Speedup: 0.28x | Eficiencia: 28.3% | Memoria: 10.6MB
Probando con 2 cores...
  Speedup: 0.35x | Eficiencia: 17.3% | Memoria: -25.5MB
Probando con 4 cores...
  Speedup: 0.26x | Eficiencia: 6.5% | Memoria: 10.9MB

Mejor configuración: 2 cores con speedup de 0.35x
Métricas JSON guardadas: C:\Buscador_Paralelo\metrics\analisis_rendimiento_20250819_190029.json
Resumen CSV guardado: C:\Buscador_Paralelo\metrics\resumen_20250819_190029.csv
Configuración guardada: C:\Buscador_Paralelo\metrics\config_recomendada_20250819_190029.txt

================================================================================
ANALISIS COMPLETO DE RENDIMIENTO DEL SISTEMA
================================================================================
PROCESADOR: Desktop Standard
   Cores disponibles: 4
   Cores recomendados: 3
   Cores óptimos: 4
   Cores máx. prueba: 8
   Recomendación: Adecuado para paralelismo
   Justificación: Para I/O + CPU intensivo, usar 3 de 4 cores evita contención. Configuración máxima de prueba: 8 cores para evaluar sobresuscripción.

------------------------------------------------------------
MÉTRICAS PRINCIPALES:
   Tiempo Secuencial: 0.10s
   Tiempo Paralelo: 0.38s
   Speedup: 0.27x (Pobre)
   Eficiencia: 6.9%
   Configuración óptima: 2 cores (Speedup: 0.35x)

------------------------------------------------------------
MÉTRICAS ESPECIFICAS DEL PROYECTO:
   Archivos/seg: 23.75
   Palabras/seg: 87,098
   Eficiencia memoria: 0.6%
   Respuesta promedio: 42ms/archivo
   Latencia inicial: 38ms
   Paralelismo máximo: 4 cores
   Rendimiento general: Mejorable

------------------------------------------------------------
RESUMEN DE TODAS LAS PRUEBAS EJECUTADAS:
   REGULAR - Paralelo-1cores: 0.28x speedup en 367ms (Memoria: 10.6MB)
   REGULAR - Paralelo-2cores: 0.35x speedup en 301ms (Memoria: -25.5MB)
   REGULAR - Paralelo-4cores: 0.26x speedup en 399ms (Memoria: 10.9MB)

================================================================================
ANALISIS COMPLETADO - 9 archivos procesados
Palabras encontradas: 33,010 total, 205 únicas
Métricas guardadas en carpeta: metrics/
```

7. Trabajo en Equipo

Descripción del reparto de tareas

CANDY - Procesamiento Paralelo

ANGEL - Modelo Probabilístico

JASON - Interfaz Web

ISMER - API y Conexión Frontend-Backend

INDI - Documentación y Métricas

Herramientas utilizadas para coordinación

- **Control de versiones**: Git/GitHub
- **Gestión de proyecto**: Azure DevOps/Trello
- **Comunicación**: Microsoft Teams/Discord
- **Documentación**: Confluence/SharePoint

8. Conclusiones

Principales aprendizajes técnicos

- Dominio de técnicas de procesamiento paralelo aplicadas a archivos
- Implementación efectiva de interfaces usuario modernas con funcionalidad drag & drop
- Optimización de rendimiento mediante análisis de métricas en tiempo real
- Desarrollo de algoritmos de predicción de texto basados en análisis especulativo

Retos enfrentados y superados

- Manejo eficiente de diferentes formatos de archivo de forma simultánea
- Implementación de sincronización thread-safe para estructuras de datos compartidas
- Optimización del balance entre paralelismo y consumo de recursos
- Desarrollo de una interfaz responsiva que no se bloquee durante procesamiento intensivo

Posibles mejoras o líneas futuras

1. **Integración con servicios en la nube** para procesamiento distribuido
2. **Implementación de machine learning** para mejorar predicciones de texto
3. **Soporte para formatos adicionales** (Excel, PowerPoint, etc.)
4. **API REST** para integración con otras aplicaciones
5. **Sistema de plugins** para extensibilidad de funcionalidades
6. **Análisis semántico avanzado** con procesamiento de lenguaje natural

9. Referencias

URL: https://learn.microsoft.com/en-us/dotnet/standard/parallel-programming/task-parallel-library-tpl

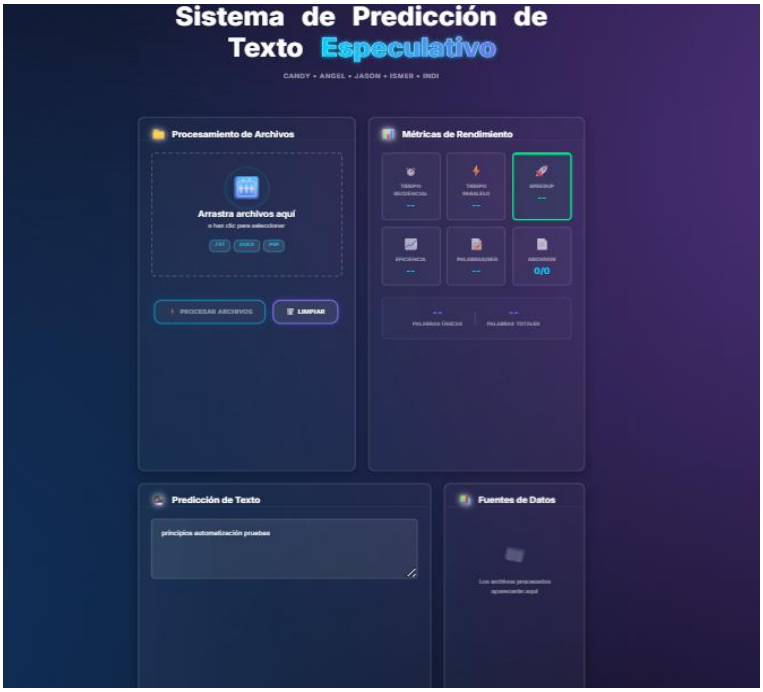
URL: https://learn.microsoft.com/en-us/dotnet/standard/parallel-programming/dataflow-task-parallel-library

URL: https://dotnettutorials.net/lesson/task-parallel-library-overview/

10. Anexos

Interfaz Principal

La aplicación presenta un diseño moderno con cuatro módulos principales distribuidos en una interfaz intuitiva.



Procesamiento de Archivos (Módulo Superior Izquierdo)

Función: Cargar y procesar archivos de texto para análisis

Características:

Zona de Arrastre: Área central con ícono de carpeta donde puedes arrastrar archivos

Texto Guía: "Arrastra archivos aquí o haz clic para seleccionar"

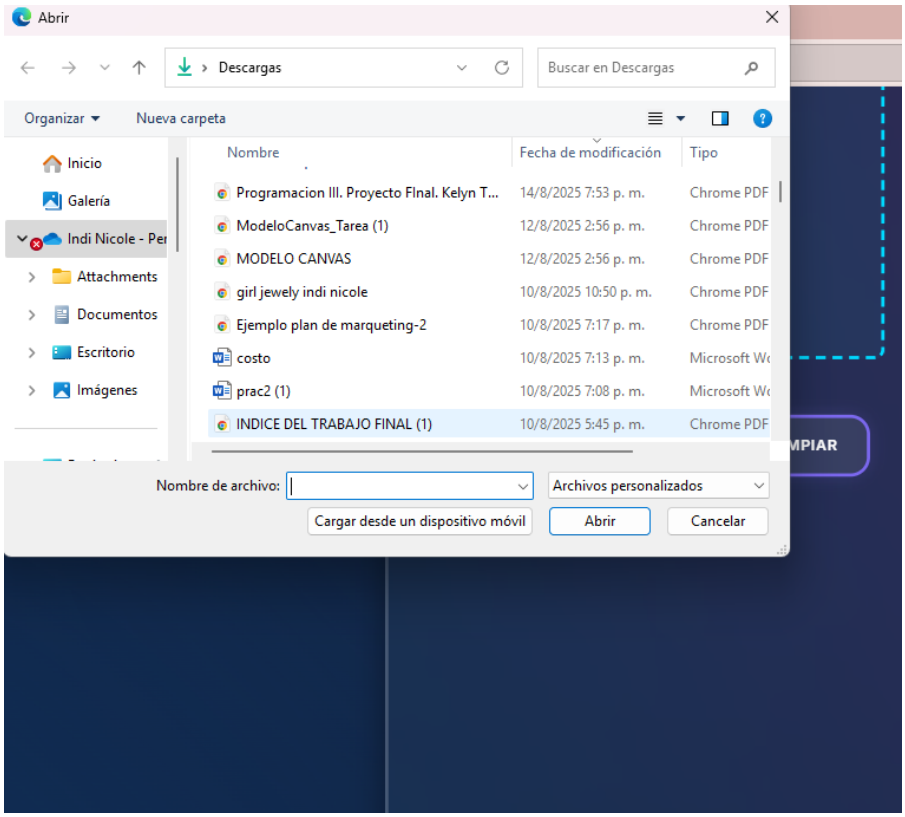
- **Formatos Soportados:** .txt, .docx, .pdf (indicados por los botones en la parte inferior)

Instrucciones de uso:

1. Arrastra archivos directamente a la zona punteada
2. O haz clic en el área para abrir el selector de archivos
3. Selecciona el formato apropiado (.txt, .docx, .pdf)
4. Presiona **"PROCESAR ARCHIVOS"** para iniciar el análisis
5. Usa **"LIMPIAR"** para resetear la selección

Paso 1: Preparar Datos

1. Accede al módulo **Procesamiento de Archivos**
2. Carga tus documentos de texto
3. Selecciona el formato correcto
4. Procesa los archivos

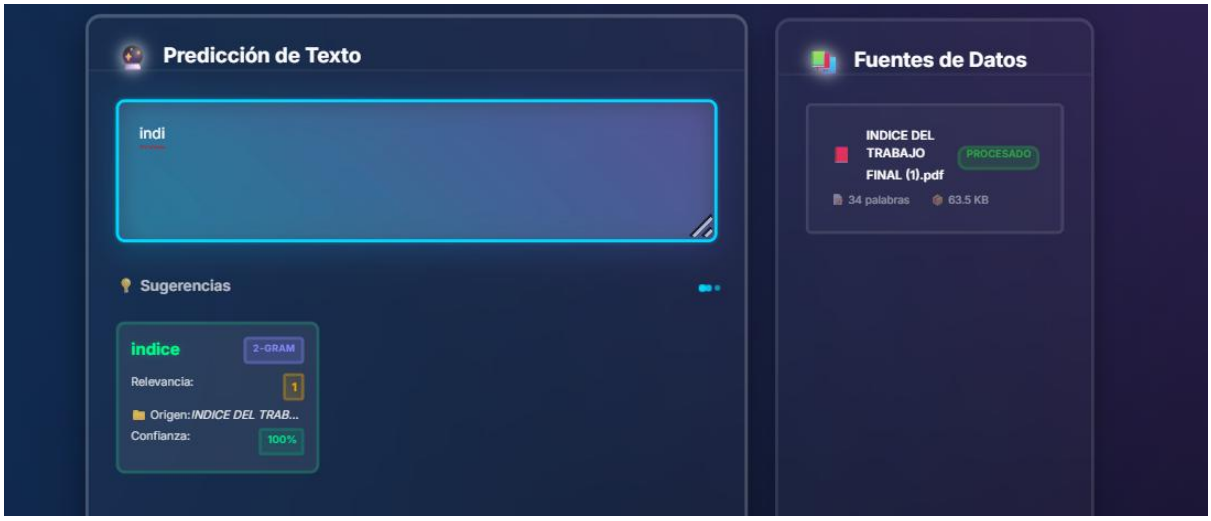


Paso 2: Monitorear Rendimiento

1. Observa las **Métricas de Rendimiento**
2. Verifica que el procesamiento paralelo esté activo (indicado en verde)
3. Compara tiempos secuenciales vs. paralelos

Paso 3: Realizar Predicciones

1. En **Predicción de Texto**, introduce tu texto base
2. Espera a que el sistema procese la información
3. Revisa las predicciones generadas



Características Técnicas

Procesamiento Paralelo

- **Optimización:** El sistema prioriza el procesamiento paralelo para mejor rendimiento
- **Monitoreo:** Métricas en tiempo real para comparar eficiencia
- **Escalabilidad:** Capaz de manejar múltiples archivos simultáneamente

Compatibilidad

- **Formatos:** Soporte para .txt, .docx, .pdf
- **Interface:** Diseño responsive y moderno
- **Usabilidad:** Drag & drop para facilidad de uso

Análisis Avanzado

- **Predicción especulativa:** Algoritmos avanzados de análisis de texto
- **Fuentes múltiples:** Integración de diferentes tipos de documentos
- **Métricas detalladas:** Seguimiento completo del rendimiento