SATIM Challenge: Policy Compliance Analysis System

Technical Report

June 14, 2025

Abstract

This technical report presents the SATIM Challenge project, a sophisticated policy compliance analysis system that leverages artificial intelligence to compare internal policies against global regulations. The system utilizes advanced natural language processing techniques, vector databases, and large language models to provide comprehensive policy analysis and recommendations. This report details the system architecture, implementation, and technical specifications.

1 Introduction

The SATIM Challenge project addresses the critical need for automated policy compliance analysis in organizations. The system provides a robust solution for comparing internal policies against global regulations, identifying gaps, and suggesting improvements. This automated approach significantly reduces the time and resources required for manual policy analysis while ensuring comprehensive coverage of regulatory requirements.

2 System Architecture

2.1 Overview

The system is built using a modern web application architecture with the following key components:

- Flask-based web server
- Policy Analyzer engine
- Vector database for policy storage
- OpenRouter API integration for LLM capabilities
- File processing system

2.2 Core Components

2.2.1 Web Application (app.py)

The web application serves as the primary interface for users to interact with the system. Key features include:

- File upload handling for PDF documents
- Text-based policy input
- Multi-language support (English and French)

- Automatic file cleanup system
- Error handling and logging

2.2.2 Policy Analyzer (policy_analyzer.py)

The Policy Analyzer is the core engine that performs the policy analysis. It implements:

- Vector database initialization and management
- Policy comparison algorithms
- LLM integration through OpenRouter API
- Structured analysis report generation

3 Technical Implementation

3.1 Vector Database Integration

The system utilizes vector databases to store and retrieve policy documents efficiently. The implementation includes:

- Separate databases for internal and global policies
- Document loading and processing
- Similarity search functionality
- Efficient document retrieval

3.2 Language Model Integration

The system integrates with Mistral-7B through the OpenRouter API to provide intelligent policy analysis. Key features include:

- Custom prompt templates for different languages
- Structured analysis output
- Error handling and fallback mechanisms

4 Features and Capabilities

4.1 Policy Analysis

The system provides comprehensive policy analysis with the following outputs:

- Missing policies identification
- Implemented policies review
- Improvement suggestions
- Best practices recommendations

4.2 Multi-language Support

The system supports both English and French languages for:

- User interface
- Policy analysis
- Report generation

5 Security and Performance

5.1 Security Measures

- Secure file handling
- API key management
- Input validation
- File size restrictions

5.2 Performance Optimizations

- Automatic file cleanup
- Efficient vector search
- Caching mechanisms
- Error handling and logging

6 Technical Requirements

6.1 System Requirements

- Python 3.x
- Flask web framework
- Vector database storage
- OpenRouter API access

6.2 Dependencies

Key Python packages required:

- Flask
- LangChain
- PyPDF2
- Requests
- Vector database libraries

7 Solution Structure

1 - Data

Collection of structured and unstructured documents from multiple sources + loading Data
techniques used: langchain, pypdf2

2- Document Preprocessing

cleaning and standardization of document formats for optimal processing efficiency **techniques used:** regex and text processing libraries

3 - Chunking & Embedding

Intelligent document segmentation and high-dimensional vector encoding for semantic understanding.
techniques used: langchain recursivecharactertextsplitter (for chunking) + openaiembedding (embedding) + faiss (to create vectordbs)

4 - Vector Database Storage

High-performance vector storage with optimized indexing for rapid semantic similarity queries
techniques used: FAISS, Qdrant

5 - Semantic Retrieval Module

Advanced semantic search identifying contextually relevant private clauses for each public document section.
techniques used: Cosine SimSimilarity

6 - LLM Analysis Engine

Large Language Model performs sophisticated comparative analysis using retrieved context and structured prompts
techniques used: LLM

7 - Gap Analysis & Reporting

Comprehensive analysis interpretation with structured findings.
techniques used: Web interface

8 Future Improvements

Potential areas for enhancement include:

- Integration with additional regulatory databases
- Advanced reporting features

9 Conclusion

The SATIM Challenge project demonstrates a robust and scalable solution for automated policy compliance analysis. The system's architecture and implementation provide a solid foundation for future enhancements and broader adoption in organizational settings.