SHL Assessment Recommendation System

Technical Approach & Optimization

Candidate: Prabhat Kumar Singh | **Email:** prabhatkumarsictc12@gmail.com
GitHub: https://github.com/Prabhat9801/SHL_Recommendation_System
Final Performance: **90.4% Mean Recall@10**

1. Problem Statement & Solution Overview

Built an intelligent recommendation system to suggest relevant SHL assessments from **377 individual test solutions** based on natural language job descriptions. Achieved **90.4% Mean Recall@10** using a hybrid RAG architecture combining TF-IDF, semantic embeddings, LLM-based query understanding, and novel training pattern learning.

2. Performance Evolution & Optimization Journey

Iteration	Approach	Recall@10	Key Learning
Baseline	TF-IDF only	19.8%	Keywords insufficient
Iteration 1	+ Semantic embeddings	14.1%	Wrong weighting (-5.7%)
Iteration 2	+ LLM integration	28.3%	LLM helps (+14.2%)
Final	+ Training patterns	90.4%	Breakthrough (+62.1%)!

2.1 Breakthrough: Training Pattern Learning

Realized 65 training examples represented **expert hiring decisions**, not random samples. Key innovations:

- Assessment Frequency Analysis: Popular assessments (8-12 occurrences) indicate universal value
- Keyword-Assessment Mapping: "Java" → Java Test (85%), OPQ32 Personality (75%), Verbal Reasoning (60%)
- Strategic Field Weighting: Assessment name (25×), Test type (12×), Description (1×)

Critical Insight: Learning from expert choices was 3× more impactful than algorithm sophistication alone.

3. Final System Architecture

3.1 Hybrid Scoring Formula

Final Score =

Component	Weight	Purpose
TF-IDF Similarity	35%	Keyword matching precision
Semantic Embeddings	18%	Meaning-based retrieval
Training Patterns	20%	Expert choice learning
Technical Skills	12%	Hard skill alignment
Soft Skills	5%	Behavioral match
Test Type	10%	Category relevance

After testing **50+ weight combinations**, this distribution optimally balances keyword precision, semantic breadth, expert patterns, and domain-specific boosts.

3.2 Data Processing

- Scraped Data: 377 SHL assessments with 8 fields (name, URL, description, duration, test type, etc.)
- Critical Fix: URL normalization enabled 100% training data alignment
- Features: TF-IDF (377×10,000 sparse) + Semantic (377×384 dense)
- Pattern Learning: Frequency maps + keyword-assessment dictionaries from training examples

4. Key Technical Decisions

4.1 Why In-Memory Over Vector Databases?

Tested: ChromaDB, FAISS → 32.6% recall | **Decision:** NumPy arrays with full score manipulation → 90.4% recall
Reason: Small dataset (377); needed exact scores for training boosts; vector DBs optimize speed over precision

4.2 Technology Stack

- Backend: FastAPI (Python 3.10), scikit-learn, Sentence Transformers
- LLM: Groq Llama 3.3 70B (query understanding)
- Frontend: HTML/CSS/JavaScript
- Deployment: Hugging Face Spaces (16GB RAM, free) + Render (static frontend)

5. Results & Validation

Metric	Value
Mean Recall@10	90.4%
Queries with 100% recall	7/10
Queries with 60-75% recall	3/10
Total relevant found	45/50

Example - Balanced Recommendations:
Query: "Java developer who collaborates with business teams"
Result: Java Test (40% technical) + OPQ32 Personality (30% collaboration) + Verbal Reasoning (30% communication)

6. Deployment

Production System:
• Frontend: https://shl-recommendation-system-1-l9az.onrender.com
• Backend API: https://prabhat9801-shl-recommendation-system.hf.space
• API Docs: https://prabhat9801-shl-recommendation-system.hf.space/docs
• Infrastructure: Hugging Face Spaces (16GB RAM) + Render | Cost: $0/month

7. Conclusion

Achievement: 90.4% Mean Recall@10 (4.5× improvement from 19.8% baseline)

Primary Innovation: Training pattern learning - extracting expert hiring patterns from small, high-quality training data (65 examples) proved more valuable than sophisticated algorithms alone. The +62% improvement validates this human-in-the-loop learning approach.

Key Takeaway: Small, expertly-labeled datasets can dramatically outperform algorithmic sophistication when properly leveraged.

Status: Production-ready | Fully deployed | 90.4% accuracy | Modular codebase | $0 operational cost