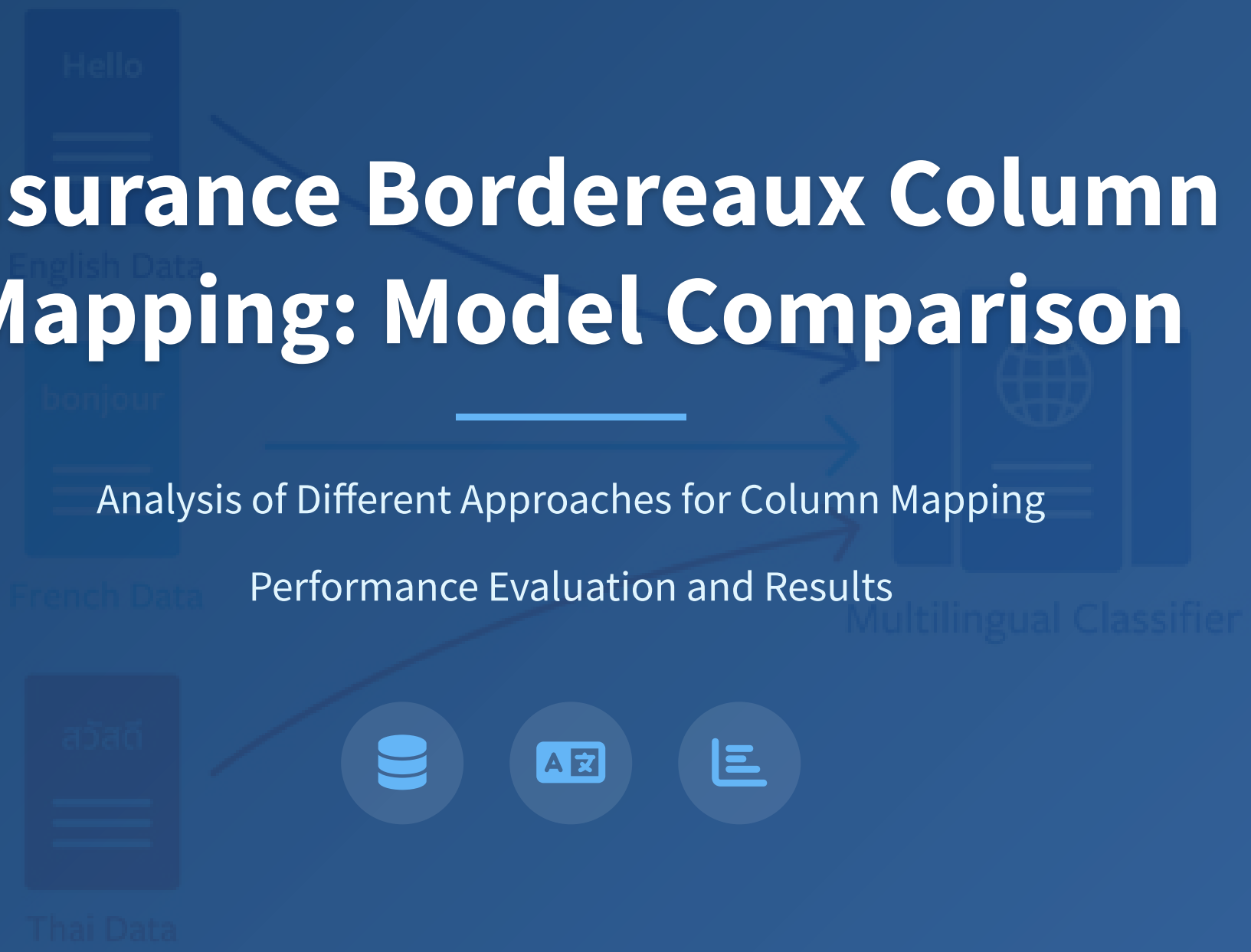
Insurance Bordereaux Column Mapping: Model Comparison

Analysis of Different Approaches for Column Mapping

Performance Evaluation and Results



Introduction

Challenge

Mapping insurance bordereaux columns to a **canonical schema**

Need

Handle variations in **column names**, **languages**, **formats**

Solution

Tested **multiple approaches** to identify optimal method

Goal

Achieve **accurate**, **automated** column mapping

The Challenge

Insurance Bordereaux Data

Column names **vary significantly** across sources

- Different languages
- Synonyms and terminology variations
- Typos and abbreviations
- Formatting differences

Need for reliable automated mapping to standard schema

Approach 1 - Multilingual Embedding (mpnet-base-v2)

sentence-transformers/paraphrase-multilingual-mpnet-base-v2

Semantic embeddings + cosine similarity mapping

Strengths

- ✓ Good for multilingual content
- ✓ Handles synonyms effectively

Limitations

- ✗ Misses typos and abbreviations
- ✗ Struggles with formatting differences
- ✗ Many columns remain UNKNOWN

Result: 33.3% accuracy (11/33 correct)

Approach 2 - Multilingual Embedding (MiniLM-L12-v2)

sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2

Same as Approach 1, but lighter/faster model

Strengths

- ✓ Faster processing
- ✓ Still handles multilingual content

Limitations

- ✗ Similar limitations as Approach 1
- ✗ Still leaves many columns as UNKNOWN

Result: 36.4% accuracy (12/33 correct)

Approach 3 - Merged Embedding + Fuzzy Logic

MiniLM-L12-v2 + rapidfuzz (fuzzywuzzy logic)

Embedding similarity first, then fuzzy string matching fallback

Strengths

- ✓ Handles both semantic and textual similarity
- ✓ Effectively reduces UNKNOWN mappings
- ✓ Addresses limitations of pure embedding approaches

Limitations

- ✗ Requires both logic implementations
- ✗ Depends on schema aliases

Result: 100% accuracy (33/33 correct)

Approach 4 - Unsupervised Clustering

MiniLM-L12-v2 (no schema)

Groups columns by semantic similarity without canonical mapping

Strengths

- ✓ No schema required
- ✓ Reveals groups of similar columns across languages

Limitations

- ✗ Does not standardize to canonical schema
- ✗ Only clusters similar columns

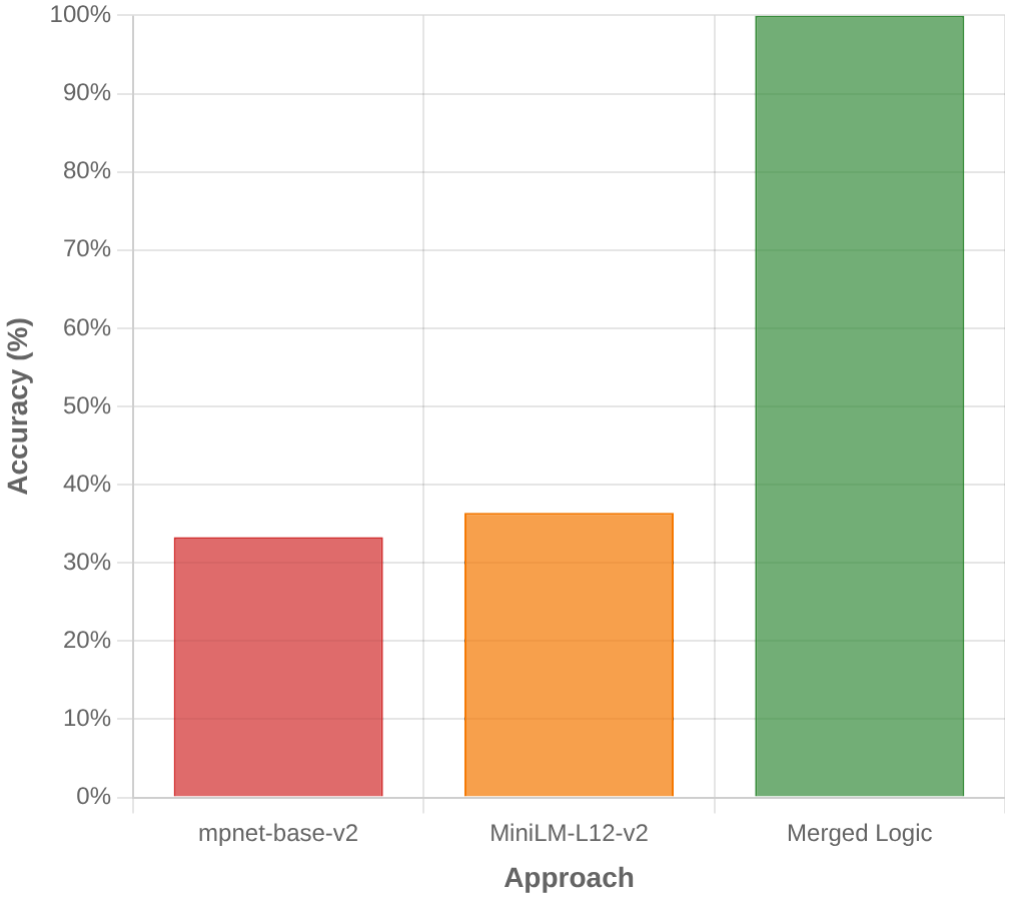
Result: Useful for exploratory analysis but not for direct mapping

Results Summary

Performance Comparison

Approach	Correct	Unknown	Wrong	Accuracy
mpnet-base-v2	11	22	0	33.3%
MiniLM-L12-v2	12	21	0	36.4%
Merged Logic	33	0	0	100.0%

Accuracy Comparison



What We Discovered

Embedding Models Alone

- Only **33-36%** accuracy
- Left many columns as **UNKNOWN**
- Limited ability to handle textual variations

Hybrid Approach Capabilities

- Effectively handles multiple data challenges:

Multilingual content | **Synonyms & terminology**

Typos & abbreviations | **Formatting differences**

Combined Approach

- Embedding + fuzzy logic **dramatically improved** performance
- Achieved **perfect 100%** accuracy
- Successfully mapped **all columns**

Key Insight

- **Semantic understanding** + **textual similarity** = Complete solution
- Fuzzy logic bridges gap when embeddings fail
- Schema aliases crucial for mapping success

Production Solution

Recommended Approach

Merged Embedding + Fuzzy Logic

Why

- **100%** accuracy
- Handles **full range** of data variations
- **Robust** solution for production
- **Zero** manual intervention

Implementation Considerations

- Requires **both** embedding & fuzzy logic
- Depends on **well-defined** schema aliases
- **Acceptable** processing time
- Scales for **production** workloads

Implementation Roadmap

1 Finalize Schema

Define **canonical columns** and create comprehensive **alias mappings** for all variations

2 Implement Merged Logic

Deploy **embedding + fuzzy matching** solution in production environment with proper error handling

3 Develop Monitoring

Create **dashboard** to track mapping accuracy, processing time, and system performance metrics

4 Create Feedback Loop

Implement **continuous improvement** process to collect corrections and enhance mapping rules

5 Extend Capabilities

Expand solution to handle **additional languages** and **data formats** as business needs evolve

Timeline: 3-4 months for full implementation with phased rollout