CV Verification System Report

2026/02/10

System Architecture and Agent Workflow

This CV verification system employs a **5-module pipeline** designed for accuracy, efficiency, and cost-effectiveness:

CV PDF

→ [Module 1] Standardization

→ [Module 2] LinkedIn Search

→ [Module 3] Profile Fetching

→ [Module 4] Two-Phase Scoring

→ [Module 5] Facebook Fallback (if needed)

Design Philosophy

- **LinkedIn-centric approach:** LinkedIn contains the most reliable professional data (work history, education, skills), making it the primary verification source
- **Hybrid matching strategy:** Pure Python for simple cases, LLM for complex semantic matching
- **Cost-aware design:** Minimize API calls while maintaining accuracy
- **Defensive verification:** Facebook serves as downside protection, not primary method

Module Breakdown

Module 1: CV Standardization

- Convert unstructured PDF text to uniform JSON format
- Implementation:
    - Use LLM to extract: name, headline, location, skills, experience, education
    - Output format aligned with LinkedIn structure for easier comparison
    - Handle missing fields gracefully (null values, empty arrays)

Module 2: Intelligent LinkedIn Search

- Find matching LinkedIn profiles with high recall
- Strategy:
    - Name-only search (no location filter) → avoid over-filtering

    - Exact headline filtering → quick first pass
    - LLM semantic matching → handle variations like "Investment Professional" ↔ "Senior Investment Analyst"
    - LLM location ranking** → prioritize geographically relevant matches
- Key Design:
    - Keep candidates with empty headlines (give second chance)
    - Only use LLM when exact match fails → cost efficiency
    - Return ranked list (best match first)

Module 3: Profile Fetching

- Retrieve full LinkedIn profiles for top candidates
- Implementation:
    - Fetch top-ranked profiles via MCP `get_linkedin_profile` tool
    - Convert to standardized format (match CV structure)
    - Add delays between API calls → rate limit protection

Module 4: Two-Phase Scoring System

- Quantify CV-LinkedIn match quality
- **Phase 1: Fast Python String Matching (No LLM)**
    - if exact_string_match(cv_field, linkedin_field):
    - score += weight
    - If score > 0.5 → Accept immediately (early stopping)
- **Phase 2: LLM Semantic Comparison (Only if Phase 1 ≤ 0.5)**
    - Batch all field comparisons in one LLM call
    - Rate limit protection: 5-second delays between profiles
    - Handles semantic equivalence (e.g., "McGill University" = "McGill")
- **Scoring Weights**

Category	Weight	Rationale
City	1	Basic verification
Country	1	Basic verification
Skills	1	CV skills averaged across all items
Experience	5 per job	Critical: company, title, seniority, dates
Education	4 per degree	Important: school, degree, field, graduation

Final Score = Total Earned / Total Possible

Module 5: Facebook Fallback (Downside Protection)

- Rescue failed verifications via alternative names
- Trigger: Only when LinkedIn score ⩽ 0.5
- Workflow:
    1. Search Facebook for CV name

2. Extract `display_name` and `original_name` from profiles
3. If alternative names found (e.g., "Alex" vs "Alexander")
4. Re-run Modules 2-4 with new name
5. Update score if successful

- Why Facebook is secondary:
  - Less structured professional data
  - Higher noise (nicknames, privacy settings)
  - Used as last resort, not primary verification

Tool Usage

MCP Tools Integration

- LinkedIn Tools (Primary)
  - *search_linkedin_people(q=name, location=None, limit=20) # Wide net*
  - *get_linkedin_profile(person_id) # Detailed info*
- Facebook Tools (Fallback only)
  - *search_facebook_users(q=name, limit=20) # Alternative names*
  - *get_facebook_profile(user_id) # Name variations*

Optimization Strategies

- API Call Minimization
  - Pure Python Phase 1: Saves ~60% of LLM calls (when score > 0.5)
  - Batch LLM queries: 1 call per profile instead of N calls per field
  - Early stopping: Accept first profile with score > 0.5

Rate Limit Management

- Between MCP calls
  - *await asyncio.sleep(1.0) # LinkedIn profile fetching*
- Between LLM calls
  - *time.sleep(5.0) # Phase 2 scoring*

Fallback Chain

LinkedIn (fast, accurate)

↓ fails

Facebook name discovery

↓

Retry LinkedIn with new name

Sample Verification Results

Test Set Performance: 5/5 Correct (100%)

```
================================================================================
FINAL SCORES (Ready for evaluation)
================================================================================
  CV_1.pdf: 0.9167
  CV_2.pdf: 0.8627
  CV_3.pdf: 0.8056
  CV_4.pdf: 0.2647
  CV_5.pdf: 0.4691
{'decisions': [1, 1, 1, 0, 0], 'correct': 5, 'total': 5, 'final_score': 1.0}
```

Detailed Verification Report

```
================================================================================
📊 CV VERIFICATION REPORT
================================================================================
Threshold: 0.5
Total CVs: 5
================================================================================

✅ Verified: 3
❌ Rejected: 2

--------------------------------------------------------------------------------


📋 DETAILED RESULTS:

[1] CV_1.pdf
    Name:       John Smith
    Score:      0.9167
    Decision: ✅ VERIFIED
    Confidence: 🟢 High Confidence

[2] CV_2.pdf
    Name:       Minh Pham
    Score:      0.8627
    Decision: ✅ VERIFIED
    Confidence: 🟢 High Confidence

[3] CV_3.pdf
    Name:       Wei Zhang
    Score:      0.8056
    Decision: ✅ VERIFIED
    Confidence: 🟢 High Confidence

[4] CV_4.pdf
    Name:       Rahul Sharma
    Score:      0.2647
    Decision: ❌ REJECTED
    Confidence: 🔴 Very Poor Match

[5] CV_5.pdf
    Name:       Rahul Sharma
    Score:      0.4321
    Decision: ❌ REJECTED
    Confidence: 🟠 Low Match

================================================================================
END OF REPORT
================================================================================
```