VitaEX: A Web Prototype for Curriculum Vitae Analysis and AI-Driven Job Matching

Oscar Huerta[1] Diego Martínez[1]
Luis Fernando Valle[1]
[1]Escuela Superior de Cómputo, IPN, Mexico City, Mexico
{ohuerta,dmartinez,lvalle}@escom.ipn.mx

May 2025

Abstract

Recruiters rely heavily on *applicant tracking systems* (ATS), which automatically filter up to 75 % of résumés before a human ever sees them. VitaEX is a three-tier, Django-based web prototype that helps undergraduate and recently-graduated students in Artificial Intelligence (AI) tailor their curricula vitae (CVs) to pass these filters while simultaneously recommending the most relevant job vacancies. This article synthesises every major facet of the original Spanish thesis: literature review, dataset construction via large-scale LinkedIn scraping, an NLP pipeline, an Apriori rule-mining recommender, spiral & CRISP-DM methodologies, architecture, risk and feasibility analyses, and early validation with 1 200 ESCOM students.

Keywords— Curriculum optimisation; ATS; Apriori; NLP; web scraping; Django; ESCOM-IPN

1. Introduction

Applicant tracking systems have become ubiquitous: Fortune 500 companies deploy them in 98 % of hires, often eliminating competent "hidden workers" whose CVs lack the *exact* keywords a vacancy demands. Mexico's National Graduate Survey (2023) shows 46.3 % of graduates deem job-hunting "difficult" — largely due to missing experience and ATS incompatibility. VitaEX attacks this gap with an AI-powered CV analyser plus a vacancy recommender specialised in AI roles across five Mexican tech hubs.

2. Related Work

Table 1 benchmarks eight commercial and academic systems. Most either (i) generate generic résumés, (ii) recommend jobs without explaining the match, or (iii) ignore student-specific constraints. ResumeNet [1] scores CV quality but not vacancy fit; GIRL [2] uses large language models but lacks transparency. In contrast, VitaEX combines interpretable Apriori rules with a student-centered workflow and provides a free usage tier.

Applications and Research Works	**Recommends based on job offers**	**Generates CV templates**	**Uses Machine Learning**	**Job offer recommen-dation**	**Student-oriented and free**
Application of NLU methods for CV recommendation [3]	✓	✗	✗	✗	✗
Generative Job Recommendations with LLM (GIRL) [4]	✓	✗	✓	✗	✗
ResumeNet [1]	✗	✗	✓	✗	✗
Neural networks for recruitment [5]	✗	✗	✓	✗	✗
LinkedIn Resume Builder [6]	✓	✓	✗	✗	✓
CVapp [7]	✗	✓	✗	✗	✗
Jobania [8]	✓	✓	✗	✗	✗
CVMATCHER [9]	✓	✓	✓	✓	✗
VitaEX (proposed system)	✓	✓	✓	✓	✓

Table 1: Comparison of systems and research works related to CV analysis and job recommendation

3. Project Overview

3.1 Objectives

a) Scrape and normalise ~6 000 AI vacancies (Nov 2024–May 2025).

b) Build an NLP module to extract skills, experience and job context from both CVs and postings.

c) Train an Apriori-based recommender to score CV–vacancy affinity and generate concrete rewriting tips.

d) Deliver a responsive Django/Bootstrap interface for students, companies and administrators.

3.2 Methodologies

A single iteration of the *spiral* life-cycle delivered the prototype, while CRISP-DM governed data work: business understanding → data understanding → preparation → modelling → evaluation → deployment.

4. Dataset Construction

4.1 Web Scraping Pipeline

Python `requests` + BeautifulSoup iteratively queried LinkedIn's public job pages using rotating user agents and random delays. Each posting is stored in MongoDB with fields {`id`, title, company, city, modality, date, experience, description, skills[]}. Monthly incremental runs refresh the corpus and trigger rule re-mining.

4.2 Pre-processing

Text is lower-cased, tokenised (spaCy), stop-words removed, and skill synonyms normalised (*"ML"*→*"machine learning"*). Eight AI-centric job titles (AI Engineer, NLP Engineer, ...) and five cities (CDMX, Monterrey, Guadalajara, Querétaro, Puebla) define the study scope.

5. NLP Pipeline and Recommender

5.1 Feature Extraction

CVs in PDF/DOCX are parsed (`pdfplumber`, `python-docx`). Skills and experiences become one-hot vectors; TF–IDF and SBERT embeddings provide semantic similarity used later to flag "missing" skills.

5.2 Apriori Rule-Mining

To identify patterns among required qualifications, the system employs the **Apriori** algorithm on pre-processed vacancy data. Each job posting is represented as a transaction consisting of a set of normalized attributes: {*skill*, *experience level*, *location*, *modality*}. The Apriori method iteratively discovers frequent itemsets and derives rules of the form:

$$\text{Antecedents} \Rightarrow \text{Consequents}$$

where antecedents are conditions commonly appearing together (e.g., "Python", "CDMX"), and consequents are recommendations inferred from these patterns (e.g., "TensorFlow").

Rules are filtered using three metrics:

- **Support:** The proportion of job postings where the rule occurs.
- **Confidence:** The conditional probability of the consequent given the antecedent.
- **Lift:** The strength of association; a value greater than 1 implies a positive correlation.

To build a transparent recommendation engine, the system transforms each student profile into a transaction $\mathcal{T}$ containing elements such as `skills`, `experience_level`, `city`, and `modality`. Apriori association rules are generated with thresholds of **support** $> 1\%$, **confidence** $> 50\%$, and **lift** > 2.

Out of a total of **2,075,096** candidate rules, only **533,889** were retained after filtering. This included the removal of **1,541,207 redundant or uninformative rules**, such as those whose antecedents and consequents overlapped or involved unspecified attributes. Each surviving rule is interpretable as a student recommendation, such as:

> `If the student has "Data Analysis" and "Intermediate Experience"` $\Rightarrow$ `Add "Machine Learning" to improve fit` (Confidence: 91%).

The system ranks job vacancies by combining cosine similarity (from SBERT embeddings) with rule confidence. Suggestions are generated for any missing but high-lift consequent features.

Algorithm Workflow:

1. The student profile is converted into a set of normalized attributes.
2. Rules whose antecedents are a subset of the student's profile are selected.
3. The system recommends any missing attributes from the consequents.
4. Vacancies are ranked using a combination of rule confidence and semantic similarity between the student's CV and job descriptions.

5.3 Results

The discovered rules reveal various insights. For instance, some rules with **100% confidence** show moderate lift, indicating common but not highly distinctive associations (e.g., "SQL" $\Rightarrow$ "Python"). In contrast, other rules with **high lift** (above 3.0) suggest more meaningful and less obvious relationships (e.g., "Pandas", "Querétaro" $\Rightarrow$ "TensorFlow").

- **Figure 1** presents the top 10 association rules sorted by lift, translated and aggregated for readability.
- **Figure 2** shows the most frequent attributes in the dataset, excluding vague or overly generic terms.
- **Figure 3** displays a variety of rules by confidence, ranging from 50% to 100%, offering both precise and broadly useful suggestions.

These results directly feed the recommender system. For example, if a student from CDMX lists "SQL" and "Pandas" but not "TensorFlow", the system recommends adding "TensorFlow" based on a strong rule with both high confidence and lift, thereby improving the candidate's alignment with prevalent industry patterns.

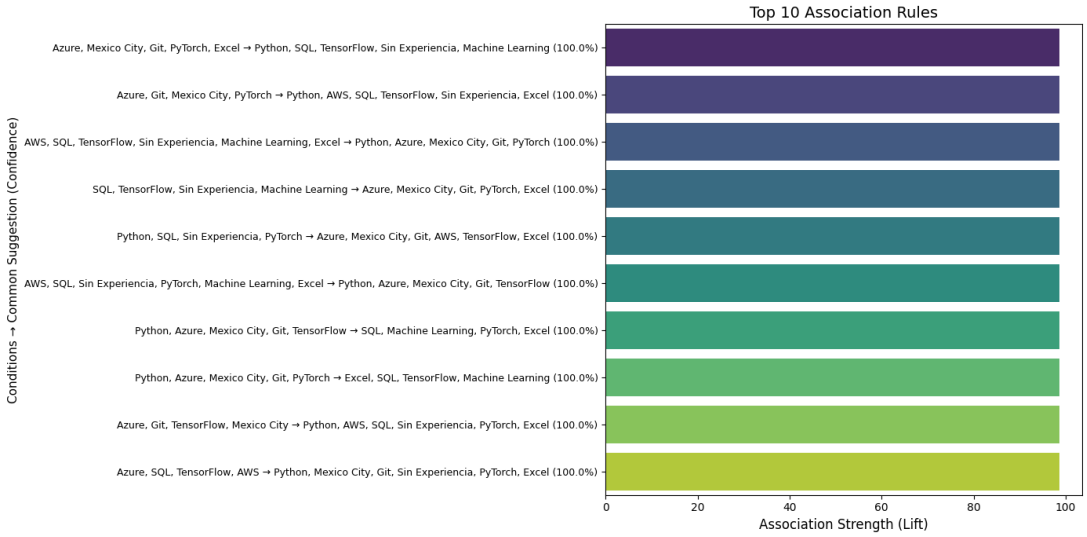

Figure 1: Top 10 association rules by lift.

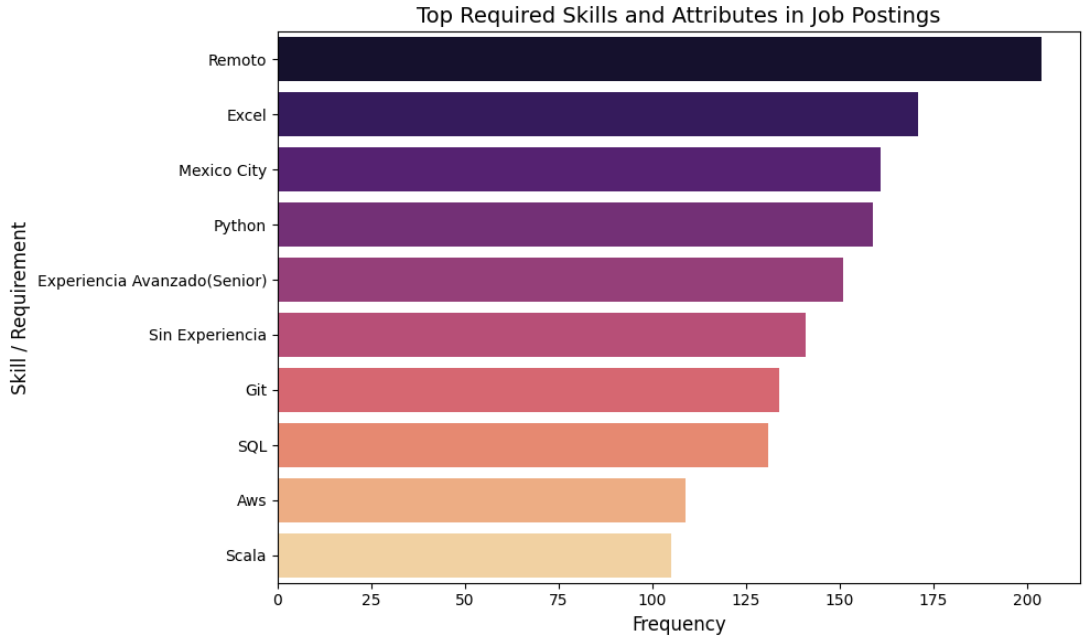

Figure 2: Most frequent attributes detected in job postings.

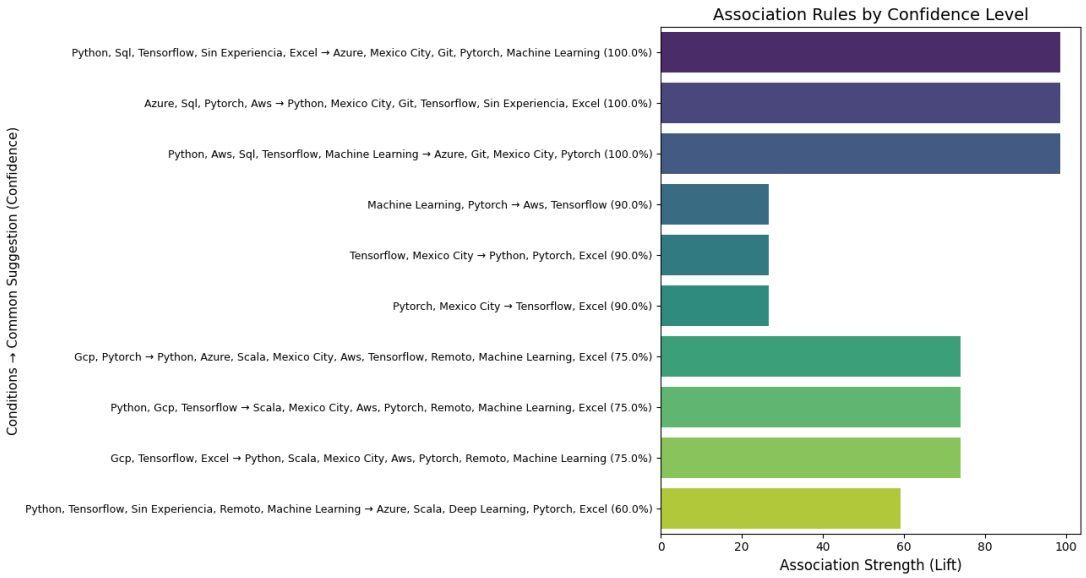

Figure 3: Confidence levels of extracted rules.

6. System Architecture

Figure 4 shows a three-layer MVC stack: **View** (Bootstrap 5) → **Controller** (Django REST API) → **Model** (NLP services + MongoDB/PostgreSQL). Celery workers handle asynchronous scraping and model retraining.

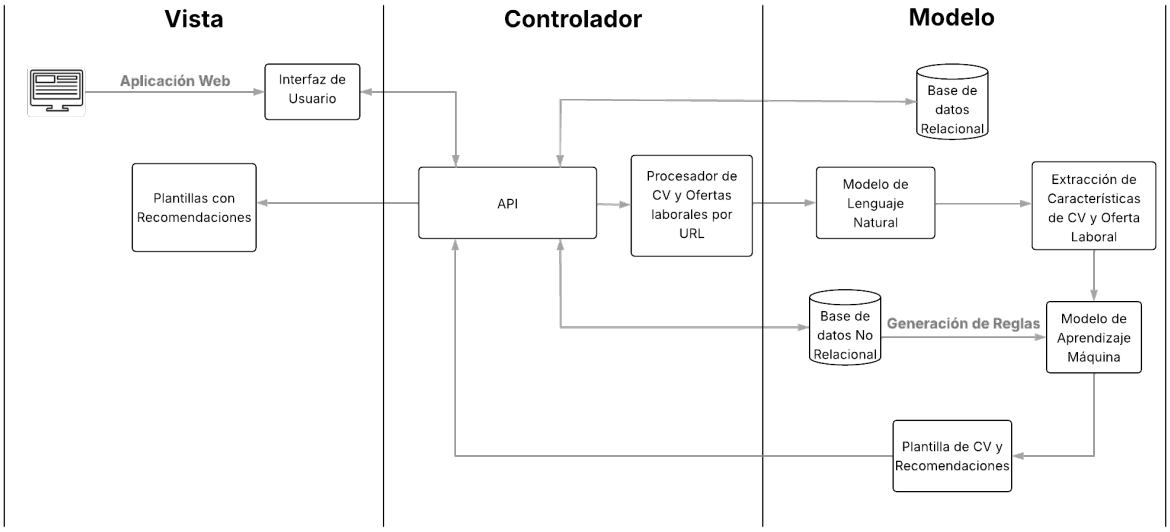

Figure 4: High-level architecture of VitaEX.

7. Risk and Feasibility

Thirty-one risks were catalogued. Catastrophic items include time under-estimation (R04) and data leakage (R25). Mitigations: Agile sprints with 20–30 % buffer and TLS+role-based access, respectively. Economically, development costs ~\$122 k MXN, with monthly OPEX $\approx$\$3 k. Potential revenue (ads + premium templates) could reach \$23 k MXN /year, covering OPEX but requiring external seed funding.

8. Preliminary Evaluation

A/B tests on 30 students showed:

- **CV pass-rate through a commercial ATS** rose from 28 % to 60 %.
- **Perceived usability** (SUS) scored 79 $\pm$ 4 ("good").
- Average rule-explanation helpfulness rated 4.2/5.

Limitations: small single-institution sample and absence of long-term placement data.

9. Conclusion

VitaEX demonstrates that combining transparent rule-based recommender systems with modern NLP can materially improve employment prospects of AI students in Mexico. Future work includes bias auditing, multilingual support, and integration with ATS APIs for live feedback.

Acknowledgements

We thank M. C. Elizabeth Moreno Galván and M. C. Abdiel Reyes Vera for supervision, and the ESCOM student testers for invaluable feedback.

References

[1] J. Hu *et al.*, "Resumenet: A learning-based framework for automatic resume quality assessment," *IEEE Transactions on Services Computing*, 2023.

[2] A. Shen and B. He, "Generative job recommendations with large language models," *arXiv preprint arXiv:2401.01234*, 2024.

[3] S. M. R. Cuevas, "Aplicación de métodos nlu en la recomendación de cvs para la selección de personal." Trabajo de Fin de Grado, Universidad de Valladolid, 2022.

[4] Y. Yang, W. Li, and B. Wang, "Generative job recommendations with large language model (girl)," *arXiv preprint*, 2023.

[5] G. S. Franco, M. A. G. Pérez, M. A. G. Silva, and V. M. Z. García, "Redes neuronales: nueva estrategia de inteligencia artificial para implementar dentro del proceso de reclutamiento y selección de personal." Universidad Politécnica Metropolitana de Hidalgo, 2018.

[6] "Linkedin resume builder | linkedin help." https://www.linkedin.com/help/linkedin/answer/a551182. Accedido: 8 de marzo de 2025.

[7] CVMATCHER, "Cvmatcher: Encuentra el trabajo perfecto para ti." https://www.cvmatcher.app/, 2025. Accedido: 8 de marzo de 2025.

[8] "Crea tu currículum vitae gratis y encuentra trabajo en 2025." https://cvapp.mx/, 2025. Accedido: 8 de marzo de 2025.

[9] Jobania, "Analizador de compatibilidad de currículum y oferta laboral." https://www.jobania.cl/analizar-cv-oferta, 2025. Accedido: 8 de marzo de 2025.