JJEE

Volume 12 | Number 3 | September 2026 | Pages 449-464

Jordan Journal of Electrical Engineering

ISSN (print): 2409-9600, ISSN (online): 2409-9619

Homepage: jjee.ttu.edu.jo

A Continuous Data-Driven Framework for Bridging the University–Industry Gap Through Emerging Skill Detection

Bilal H Hawashin[1*], Eyad M Hailat[2]

[1, 2] Al-Zaytoonah University of Jordan, Jordan
E-mail: b.hawashin@zuj.edu.jo

Received: Feb 21, 2026 *Revised: May 15, 2026* *Accepted: May 16, 2026* *Available online: Aug 15, 2026*

***Abstract*—** Bridging the gap between industry and university is one of the major challenges facing both sides in recent years due to the rapid digital transformation and emerging technologies. Despite having several proposed works in this direction, most of these works concentrated on fixed comparisons and periodic surveys without continuous mechanisms to capture rapidly changing industry requirements. This work proposes a continuous data-driven framework that can analyze recent job posts and provides the current and emerging industrial needs in terms of skills, competences, and job roles. These results would be delivered to a dynamic interactive dashboard that allows universities to periodically update their curricula, learning outcomes, and other activities in accordance with industrial needs. Accreditation programs and individuals can benefit from the feedback as well. Experimental results showed high alignment between the extracted skills and current national skill reports. Our proposed framework outperformed baselines and captured emerging trends.

***Keywords*—** Bridging gap between universities and industry; Job post analysis; periodical alignment; Emerging industrial needs.

1. INTRODUCTION

Bridging the gap between universities and industry refers to the alignment between university outcomes and industrial needs. This alignment would reduce the gap between what has been taught in universities in general and what is expected by the industry. On the one hand, Industry is changing rapidly and always seek state-of-art tools and efficient mechanisms to achieve the highest level of competitiveness possible, which would be linked directly to profit. On the other hand, universities concentrate on spreading knowledge, paying more attention to theoretical fundamentals, original works, and how to convey the knowledge in clear means to students. They pay attention to skills and competences, and the national qualifications framework, which has been adopted by several countries including Jordan, is an attempt to insist on the importance of including not only knowledge but also skills and competences in the educational system. Despite these efforts, the need for further collaboration between universities and industry is still essential.

Several works in the literature have proposed solutions to this challenge. However, these works mostly included high level surveys of the challenges and future suggestions without providing continuous concrete alignment method. It is necessary to have a well-established, clear, and continuous mechanism that implements the outcomes of the collaboration. Such mechanism would help several parties. First, graduate students seeking jobs. These students suffer from not being totally equipped with the current necessary knowledge and skills needed

* Corresponding author
DOI: https://doi.org/10.5455/jjee.204-1771677153

in the industry. This would result in making it harder for them to find a job. Even when they find one, in most cases they are required to train and spend time and effort to adapt to industry needs. Second, industry. This side also suffers in the process of finding a suitable graduate that meets their specific needs. Third, universities. This side is equally important as they play the major role in preparing students to meet industry requirements.

In our work, we propose a continuous data-driven framework that bridges the gap between universities and industry. This framework has many functionalities and would serve graduates, universities, industry, accreditation process, and other parts. One of the key functionalities in this platform is its ability to extract job posts in a specific time interval from well-known platforms such as Linkedin. Later, the extracted job posts would be processed to get the trending skills. The trending skills are skills that appeared in job posts increasingly in recent years. Finally, the trending skills are used to periodically update a reference dashboard for this sake. Universities and other stakeholders can consult the dashboard to obtain insights on the up-to-date industrial needs.

As for universities, the use of these extracted skills can be made by different means. For example, extracurricular training done inside universities can be directed toward these skills. Graduation project advisors would encourage students to select on these skills. Training course or bootcamp course can emphasize on these skills as well. Moreover, some programs have courses named "selected topics". For example, "Selected Topics in Artificial Intelligence". This course can cover part of the trending skills, which are actual industrial needs, instead of giving advanced topics regardless of their actual need to the industry. Furthermore, many universities have started to conduct student competitions, which is a great mean to sharpen skills via competitions. However, the topics of these competetions can also benefit from the extracted trending skills. Finally, students would be encouraged to take elective training in these topics. As a result, students would be prepared to the industrial needs while they are still in the university, and they will be prepared with the topics that currently matter the most.

As can be noted, trending skills would serve as directed insights that can be used by universities to cover what is really in need. As a result, graduates would be better equipped with the industrial needs. This process can be performed periodically, and as a result, the trending skills would change from time to time to reflect what is really needed. It would act as a dynamic dashboard to the industrial needs currently.

As for individual students, they can periodically consult the dashboard to get current industrial needs and plan to address it by extracurricular activities such as training and certification programs.

Regarding accreditation, they would equally benefit from such extracted skills as they represent evidence of the current industrial needs that can support continuous program reviews, learning outcomes mapping, and stakeholder engagement, which are very important in the program accreditation process.

The contributions of this work are as follows:

- Developing a data-driven framework that automatically extracts recent job posts and analyzes them to extract the emerging industrial needs.
- Introducing a dynamic pipeline to support periodical update of university curricula, learning outcomes, and training activities in accordance with the extracted industrial needs, which would enable sustainable alignment between universities and industry.

The rest of this paper is as follows. Section two is for the literature review. Section three provides a comprehensive description of the proposed platform. Section four provides experimental results, and section five provides the conclusions and future work directions.

2. LITERATURE REVIEW

Several works have studied this topic before. Freitas, Marques, and Silva (2013) showed the importance of collaboration between universities and industries. Furthermore, they used data from 24 research groups in science and engineering departments in Brazil to show that such collaborations are affected by the degree of industry matureness. They showed the reasons behind the varying levels of collaboration and why tailored strategies are necessary.

Tambe, P. (2014) showed that technology investment alone is not enough for firms. The need to have employees with the right skills and competencies is equally important. This shows the importance of having universities providing skilled graduates with no education gaps with the industry. As an example, companies that used between Hadoop between 2006 and 2011 experienced about a 3% increase in productivity growth, only when firms already obtained data resources that could be used and when fostering a local workforce with the needed skills.

Ankrah, S., & Al-Tabbaa, O. (2015) defined university-industry collaboration and reviewed recent works in this field. They categorized collaborations into several types, and provided insights of the barriers that make such collaborations more difficult. Finally, they provided suggestions to solve these issues. They argued that the gap is not only In skills but also in institutional misalignment and coordination challenges.

Perkmann, M., & Schildt, H. (2015) suggested the use of open data partnerships instead of protecting the results of the collaboration via patents and keeping it secret. It also showed the role of the boundary organizations and how they can be used to attenuate the differences in objectives between industries and universities. It is know that industries seek profit by making new discoveries and keeping them secret, while universities seek open knowledge. This conflict can be attenuated using such organizations that can provide solutions that are acceptable for both parties. They provided a case study that proved their argument.

Jackson, D. (2015) argued on the importance of bridging the gap between universities and industry by using previous employer studies and literature to get industry needs first, then integrating employability skills in the learning process. The work insisted on the importance of concentrating on skill and competence development inside universities to provide ready graduates.

McLaughlin, J. E., Lupton-Smith, C., & Wolcott, M. D. (2018) suggested the use of text mining to find industry needs and to check its suitability with what universities provide. As part of the process, they analyzed several job postings to get the required skills. They insisted on the importance of using text mining in obtaining useful knowledge forming industrial needs.

Rybnicek, R., & Königsgruber, R. (2019) studied the topic of collaboration between universities and industry and concentrated on several factors that could contribute to succeeding such collaboration. These factors include organizational and institutional factors, human and relational factors, knowledge and research factors, and finally contextual and environmental factors. They concluded that the integration of these factors is vital for successful collaborations.

Sjöö, K., & Hellström, T. (2019) argued that collaborations between universities and industry is not only one type and can be categorized based on several factors. They insisted on the importance of having alignment in the goals between both parties. Also, they argued that the organizational and institutional context is important to endure such collaboration. Almaleh, A., Aslam, M. A., Saeedi, K., & Aljohani, N. R. (2019) made a cross-sectional study to bridge the gap between university curriculum and needed market skills. In their work, they used Naïve Bayse to categorize job posts. Later, similarity is used to find the gap between market needs and existing university curriculum. While our approach analyzes job posts, it tends to detect continuous changes in market needs as the process is continuous. Furthermore, our study concentrates on recent year job posts. This would make it a dynamic trend-oriented analysis study. Clemente-Mediavilla and Antolín-Prieto (2019) conducted a descriptive content analysis by identifying the needed graduate skills within a specific national context (Spain). Their study categorized and provided both the technical and soft skills needed at a specific time. Cohen, M., Fernandes, G., & Godinho, P. (2025) focused on collaboration impact measurement. They proposed methods to calculate the outcomes and evaluate the results. Furthermore, their analysis included multi stakeholders such as universities, firms, and governments. They also identified current gaps in measurement methods and called for novel data-driven approaches. Du Plessis, M., Barkhuizen, N., & Schutte, N. (2024) highlighted how collaboration between universities and industry reshaped the employability requirements and strategies after covid-19. Specifically, they insisted on the hybrid and the digital competences as core outcomes. Furthermore, they showed that experiential learning, internships, and real-world projects have become vital in such university industry collaborations. Chen, Y., Liu, X., & Zhang, W. (2025) linked the university-industry collaboration with increase in enterprise productivity. In details, these agreements improve human capital, increase innovation, and lead to better managerial practices. Furthermore, they argued that this collaboration would commonly have broader economic and industrial developments. Bu, L., Li, Y., & Wang, Z. (2025) linked the collaboration between universities and industry with the green and sustainable innovation outcomes. They argued on the importance of social capital in improving the ability of firms to innovate in such sustainable environments. They proved also that collaboration would impact environmental transformation. Khan, S., Ahmed, R., & Malik, M. (2025) focused on dual-stakeholder perspective; students and firms. Furthermore, they argued on the importance of industry involvement in university curriculum development and student evaluation. They provided evidence that real-world projects and transferable learning improve the level of readiness for graduates more than merely concentrating on conceptual and survey-based frameworks.

3. METHODOLOGY

In this section, we provide the details of the proposed framework.

3.1. System Architecture

Figure 1 illustrates the layered architecture of JobBridge as a structured and modular stack that delineates user interaction, application orchestration, intelligent modeling, and data persistence. At the highest level, the Presentation Layer provides the web-based interface through which users access dashboard analytics, job search and exploration functionalities, graduate–employer matching mechanisms, administrative upload tools, and external data

ingestion controls. User interactions are translated into HTTP requests that are processed by the API Layer, implemented using FastAPI routers. This layer is responsible for request validation, parameter parsing, endpoint exposure (e.g., jobs, matching, statistics, administration, scraping), and controlled delegation of execution to internal services.

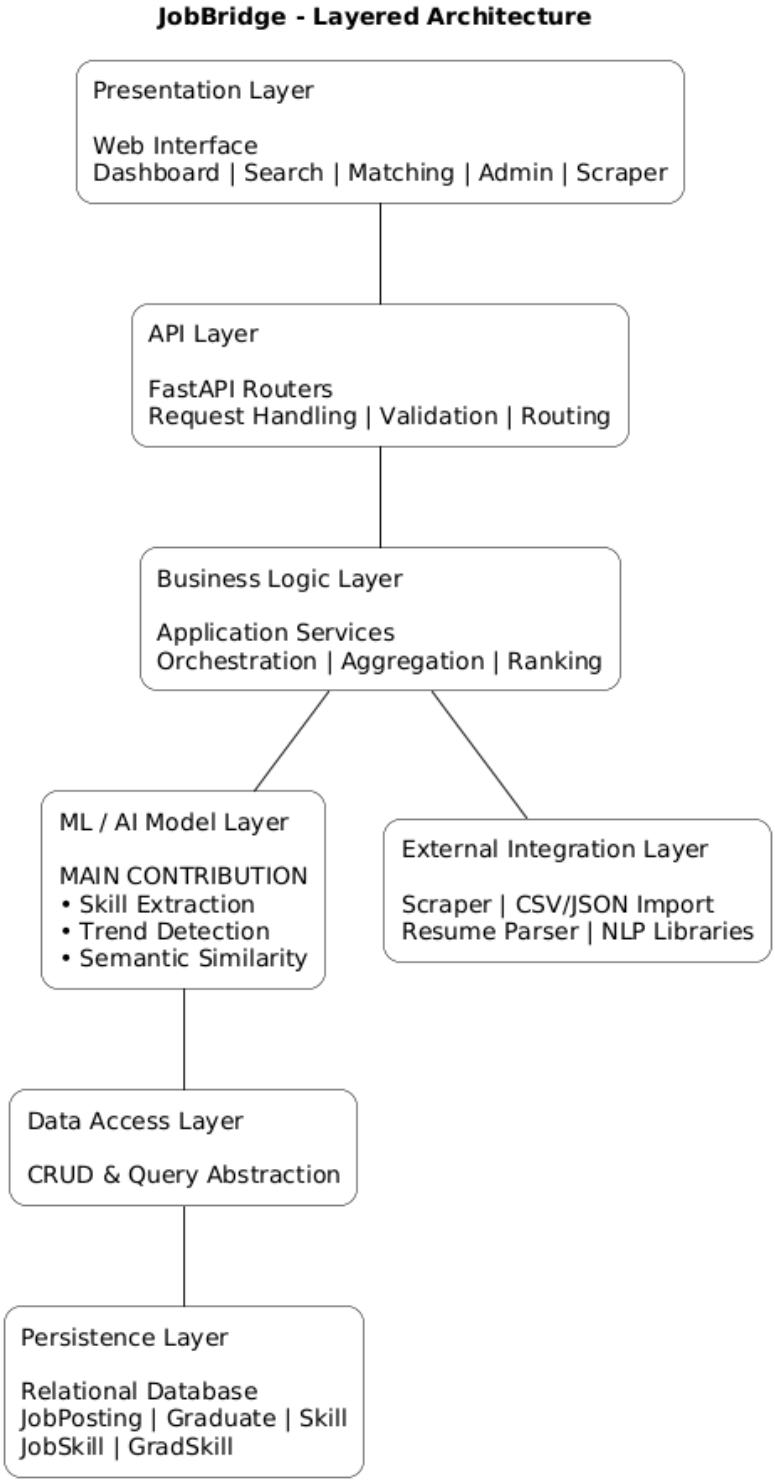

Fig. 1. JobBridge architecture.

The Business Logic Layer coordinates application workflows, including job ingestion, filtering, ranking, aggregation, and response construction. This layer orchestrates interactions between the API interface, the modeling components, and the persistence mechanisms. Central to the architecture is the ML/AI Model Layer, which constitutes the principal methodological

contribution of the system. This layer performs (i) lexicon-based skill extraction from unstructured job descriptions and graduate profiles, (ii) temporal trend and demand detection through time-windowed aggregation and statistical analysis, and (iii) semantic similarity computation for ranking and matching entities. These processes transform unstructured textual input into structured skill representations and quantitative similarity scores that drive the system's recommendation and analytics capabilities.

Data persistence is managed through the Data Access Layer, which encapsulates CRUD operations and abstracts relational queries from higher-level services. The underlying Persistence Layer is implemented as a normalized relational schema (SQLite in the proof-of-concept implementation) comprising core entities—JobPosting, Graduate, and Skill—and explicit many-to-many relationships (JobSkill, GradSkill). This schema supports referential integrity, reproducible matching operations, and efficient aggregation queries required for trend analytics and curriculum gap analysis.

From a temporal and operational standpoint, the system incorporates a scheduled data ingestion pipeline. A Python-based cron process executes daily at a predefined time and retrieves job postings published within the preceding 24-hour interval. During this batch ingestion phase, normalization procedures and deduplication checks are applied to ensure that duplicate records are not introduced into the database. Deduplication is enforced through a combination of uniqueness constraints and idempotent insertion logic (upsert semantics), thereby preserving data integrity across repeated executions. This scheduled batch process operates alongside real-time ingestion pathways (e.g., employer submissions and administrative uploads), resulting in a hybrid update strategy that balances immediacy and systematic refresh cycles.

Database interaction is implemented via a Python Object-Relational Mapping (ORM) framework, which abstracts database-specific syntax and operations from application-level logic. Consequently, migration between relational database engines (e.g., SQLite for local experimentation and PostgreSQL or MySQL for production-scale deployment) can be achieved without modification to the business or modeling layers. This architectural abstraction enhances portability, scalability, and maintainability.

Although the proof-of-concept deployment relies on a local development environment, the architecture is designed to accommodate industry-scale infrastructure. For instance, Elasticsearch may be integrated to support distributed full-text indexing, advanced search capabilities, and scalable query execution. Similarly, a data lake architecture (e.g., object storage combined with distributed processing frameworks) can facilitate large-scale historical data retention, longitudinal labor-market analysis, and model retraining workflows. The ML/AI layer is compatible with standard industrial tooling for feature management, vector storage, and model deployment, enabling systematic evolution toward production-grade AI pipelines without fundamental architectural modification. External job data acquisition, including LinkedIn job listings, is performed via a commercial data API service rather than through custom-built web scraping infrastructure. This design decision is motivated by methodological, legal, and operational considerations. Commercial APIs typically ensure compliance with platform terms of service and applicable data governance frameworks, thereby mitigating legal and ethical risks. Furthermore, API-based access provides structured and stable data interfaces, reducing susceptibility to front-end structural changes that frequently disrupt traditional scraping mechanisms. The use of managed data services also enhances reproducibility,

reliability, and maintainability by providing standardized schemas, rate management, and service-level guarantees. Importantly, this approach allows the research effort to concentrate on the system's primary scientific contribution—skill extraction, trend analysis, and semantic similarity modeling—rather than on the maintenance of fragile scraping infrastructure. In summary, JobBridge operates through a coherent multi-layered architecture in which data enters via user submissions or scheduled ingestion processes, is processed and enriched by the ML/AI layer under application-level orchestration, persisted through an abstracted data access layer, and ultimately delivered to end users via structured API responses rendered in the presentation interface. Algorithm 1 below presents our proposed trending skill extractor. The algorithm is described in details in the following subsections.

Algorithm 1. The Trending Skills Extractor

Input:
- DataSource : { "scrape" | "import_csv" | "import_json" | "post_form" }
- ScrapeParams : (keywords, location, date_range, limit) [if DataSource = "scrape"]
- TimeWindowDays : integer (e.g. 365)
- TopK : integer (e.g. 20)
- LexiconPath : path to skill_lexicon.csv

Output:
- TrendingSkills : list of (skill_name, count) sorted by count descending, length ≤ TopK

```
LEXICON ← LoadSkillLexicon(LexiconPath)
RAW_JOBS ← ObtainJobPosts(DataSource, ScrapeParams)
For each job in RAW_JOBS:
    TEXT ← PreprocessJobText(job)
    SKILLS ← RecognizeSkillTerms(TEXT, LEXICON)
    PersistJobAndSkills(job, SKILLS)
PAIRS ← GetSkillDatePairsFromDB()                    // (skill_name, posted_at) for all job_skill links
PAIRS_IN_WINDOW ← FilterByTimeWindow(PAIRS, TimeWindowDays) // keep only posted_at in last TimeWindowDays
COUNTS ← AggregateCountsBySkill(PAIRS_IN_WINDOW)     // count occurrences per skill name
TrendingSkills ← TopKByCount(COUNTS, TopK)
RETURN TrendingSkills
```

3.2. The Pipeline of the Proposed Framework

In this subsection, the pipeline stages are presented in details. These steps include data scrabbing, data preprocessing and normalization, Anonymization and privacy protection, trend detection, and dashboard update.

3.2.1. Data Scrabbing

The job data collection component is implemented as an automated procedure, referred to as ScrapeLinkedInJobs, which retrieves job postings through a commercial job-search API. The algorithm takes as input a set of keywords, a target location, optional date limits, and the maximum number of jobs to be retrieved. Based on these inputs, the system builds a search request and applies pagination while respecting the API constraints. If time boundaries are provided, the request is restricted to job postings published within the specified period. The system then sends an authenticated HTTP request and receives the results in JSON format. These results are parsed and converted into a unified internal structure containing the job title,

company name, description, posting date, and source link. Each record is subsequently checked, cleaned, and normalized to ensure consistency before being forwarded to the modules responsible for skill extraction, trend analysis, and matching within the JobBridge platform. The scrapper algorithm Is provided in Algorithm 2.

Algorithm 2. **Job Posts Scrabber**

Input:
keywords (string), location (string), start_date, end_date (ISO date or null), limit (integer)
Output:
list of job objects, each with title, company_name, description, posted_at, url, etc.

1. Build request to external job-search API (e.g. LinkedIn Job Search API):
   - Query parameters: keywords, location (e.g. "Jordan"), count = min(limit, API_MAX), start = 0
   - Optional: date filters (listedAt, listedAtEnd) from start_date, end_date
2. Send HTTP GET with authentication (e.g. Bearer token)
3. Parse response JSON into a list of job objects
4. For each job in the list:

3.2.2. Data Preprocessing and Normalization

For this framework, we apply a lexicon driven preprocessing pipeline that is designed to preserve the original job text while making skill detection reproducible:

- Encoding & basic cleaning: All CSV/JSON/resume inputs are read as UTF 8. We trim leading/trailing whitespace, normalise internal whitespace (collapse repeated spaces/newlines), and drop obviously empty rows (e.g. missing both title and description).
- Field normalisation: For imported jobs we coerce dates into ISO 8601 (YYYY MM DD or full ISO datetime), parse numeric fields (e.g. grad_year), and normalise locations to a single free text location field. Employment types coming from lists/arrays are normalised to a comma separated string over a small allowed set.
- Case and tokenisation: We do not permanently lowercase the stored job text; we preserve original casing for display. For skill extraction, however, we tokenize the concatenated title + description using NLTK (with a regex fallback) and convert tokens to lowercase. Lexicon terms are also compared in a case insensitive way, so "Python", "PYTHON", and "python" are treated identically for matching.
- Similar-word and phrase unification: We don't run a general stemmer/lemmatizer; instead, unification is driven by the curated skill lexicon. The lexicon lists one canonical form per skill (e.g. "machine learning"), and we sort it by decreasing length so multi word phrases are matched before their substrings (e.g. "machine learning" before "learning"). If the lexicon contains both a broad and a specific term, we respect that; we don't invent new merges beyond what the lexicon specifies.
- Robust matching: Single word skills must appear as full tokens (so "R" only matches the token r, not inside "engineer"); multi word skills must appear as consecutive tokens; skills with punctuation (e.g. "C#", "Node.js") are matched with word boundary regexes so they aren't picked up inside longer identifiers. Duplicates are removed per job, so each skill appears at most once per posting.

Overall, the job text itself is minimally normalized (to avoid losing nuance), while skills are normalized strongly via the lexicon and lowercase tokenization, which is what the trending skills analytics is built on.

3.2.3. Anonymization and Privacy Protection

From a privacy perspective, the system is designed to operate primarily at the job and organization level, not at the individual person level:

- Job postings: We store job title, company/organization name, location, description, and URLs, but we do not require or persist recruiter names, personal email addresses, or phone numbers as separate fields. For external feeds, we keep only the fields needed for research (title, organization, location, description, posting dates, URLs) and discard extraneous metadata where possible.
- Graduates: For the internal graduate dataset, we do store graduate level identifiers (name, email, etc.) because matching is done per graduate. However, for analysis and publication (e.g. trending skills, curriculum gaps) we only use aggregated skill counts and program/major information; no individual graduate identifiers are included in the analytics outputs.
- Resumes: When resumes (DOCX/PDF) are uploaded, we parse them to extract structured fields (name, email, phone, skills) and store only those structured fields in the graduate and grad_skill tables. The raw file content is not used in analytics and can be discarded or stored in restricted locations; logs do not contain the raw resume text.
- Anonymity in statistics: All "trending skills" and dashboard statistics are computed at skill and job count level, not at person level. When exporting or presenting results, it is straightforward to drop direct identifiers such as URLs or external IDs and to report only aggregated counts per skill, program, or time window.

3.2.4. Trend Detection

Skills are accumulated in two steps: (1) extraction from job text, and (2) storage in the database. For each job (from scraping, bulk import, or the post-job form), we concatenate title and description, tokenize the text, and match it against a curated skill lexicon (token- and sequence-based, case-insensitive). Each recognized term is stored once per job in a job–skill link table (with an optional weight). The lexicon is shared across all jobs, so the same skill name (e.g. "Python") is linked to many jobs, giving a single canonical skill set. Skills are therefore accumulated by repeated extraction from new jobs and upsert into the skill table plus insertion of job_skill rows; no manual tagging is required.

Statistics are computed from these stored links. We take all job–skill pairs and join them with the job's posted_at date. We then restrict to a time window (e.g. the last 365 days). Within that window, we count how many times each skill name appears (each job–skill link counts once). Skills are ranked by this count and we report the top-k (e.g. top 20). This would act as the skill score. The system exposes this via an API (e.g. GET /api/skills/top?window_days=365&limit=20), which the dashboard uses to show "Top Skills (last year)". The frequency score can indicate skill demand within a time interval. Finally, based on these frequency scores, the skill growth can be computed. This would indicate emerging and trending skills.

3.2.5. Dashboard Update

The output of this algorithm can be used to update the dashboard. The dashboard would act as the reference used by universities and individual students to be aware of the current

industrial needs . It also serves as the system entry point. It presents aggregate statistics including total jobs, total graduates, recent job counts, top skills (configurable time window), and language distribution of job descriptions. These analytics provide real-time labor market signals.

As for the universities, such skills can be vital to allow universities to update the content of their courses, specify the important topics that needs further training sessions, and show the best directions to select from by graduation project students. As for individual students, these skills can provide to the current industrial needs. This would be of a great help when selecting what training session to take. As stated previously, the skill extractor algorithm can be used periodically to update the dashboard with the current industrial needs, and the dashboard would act as the state-of-art reference point that directs universities to the correct skills and competences to develop.

4. EXPERIMENTS

4.1. Dataset Description

In our approach, we adopted the use of linkedin as a source of job posts due to its widely use in this regard. As described in the methodology, The jobs are extracted periodically to update the dashboard.

The following fields are extracted and stored for each job record in the job_posting table:

- job_id (INTEGER): Primary key, automatically generated.
- external_id (VARCHAR): External identifier used for deduplication (e.g., LinkedIn ID or import reference).
- title (VARCHAR): Job title.
- company (VARCHAR): Company or organization name.
- location (VARCHAR): Job location (city, region, or country as a string).
- source (VARCHAR): Ingestion source (e.g., csv_seed, json_import, post_job_form, linkedin).
- posted_at (DATETIME): Date and time the job was posted.
- url (VARCHAR): URL of the original job posting.
- raw_text (TEXT): Full job description text.
- standardized_title (VARCHAR, optional): Normalized job title for analytical consistency.
- normalized_location (VARCHAR, optional): Normalized location field for aggregation and filtering.
- valid_through (DATETIME, optional): Expiration date of the job posting.
- external_apply_url (VARCHAR, optional): External application link.
- employment_type (VARCHAR): Comma-separated employment types (e.g., FULL_TIME, PART_TIME).
- language (VARCHAR): ISO 639-1 language code of the job description (e.g., en, ar), automatically detected.

In addition, skills are extracted from the title and raw_text fields using a curated skill lexicon. Extracted skills are stored in the job_skill association table, which contains (job_id, skill_id, weight) to represent the relationship between jobs and their corresponding skills. Canonical skill names are maintained in the skill table to ensure normalization and consistency across all job and graduate records.

The experimental dataset consists of 730 job postings collected in Jordan over a one-month interval. Data acquisition targeted job listings associated with the Jordanian labor market (country code: JO), without restricting the dataset to specific job domains or occupational categories. Consequently, the dataset reflects a heterogeneous cross-section of roles spanning multiple sectors. The initial data collection covered a one-month query window. Following this initial extraction phase, the ingestion pipeline transitioned to a scheduled daily update process (beginning after February 15, 2026), in which newly posted jobs from the preceding 24-hour interval were incrementally added to the database. At the time of reporting, the total number of records obtained through the LinkedIn data source was 730 entries.

Regarding annotation, no manual labeling or human-in-the-loop tagging was performed after data ingestion. Instead, the system applies automated enrichment procedures immediately upon scraping or importing structured files (JSON or CSV). These enrichment steps include lexicon-based skill extraction from job titles and descriptions, automatic language detection (ISO 639-1), and, when available, normalization of employment type metadata. This process ensures consistent structured representation while maintaining full reproducibility and eliminating subjective bias introduced by manual annotation.

4.2. Hardware and Software Settings

The system was developed and evaluated in a standard computing environment without reliance on specialized hardware accelerators or distributed infrastructure. All experiments and system execution were conducted on conventional development machines equipped with modern multi-core processors (x86_64 or ARM64 architectures). The backend operates as a single-process ASGI application, with concurrency managed by the Uvicorn server.

The minimum memory requirement for stable execution is approximately 2 GB of RAM; however, 4 GB or more is recommended when running the backend and frontend concurrently, particularly during bulk CSV/JSON ingestion or resume processing. Persistent storage is provided through a local disk-based SQLite database file (jobbridge.db), along with temporary storage for uploaded documents and seed data (e.g., the skill lexicon and sample datasets). For proof-of-concept scale (approximately 730 job records), storage requirements remain within tens to hundreds of megabytes.

Network connectivity is required only for serving the frontend and API, as well as for accessing optional external job data sources via commercial APIs. All core functionalities, including CSV/JSON import, skill extraction, matching, and analytics, can operate fully offline.

4.2.1. Backend Implementation

The backend is implemented in Python 3.11+ using the FastAPI framework (≥0.111.0), served via Uvicorn (≥0.30.0) as an ASGI application. Data persistence is managed using SQLite 3, accessed through SQLAlchemy (≥2.0.30) as the ORM layer. Request and response validation, as well as configuration management, are implemented using Pydantic (≥2.7.0) and Pydantic-Settings (≥2.4.0).

The system employs a lexicon-based skill extraction mechanism implemented in backend/app/services/extract.py, leveraging NLTK (≥3.8) for tokenization (e.g., word_tokenize with Punkt models). Automatic language detection is performed using langdetect (≥1.0.9), with fallback heuristics when required. Resume ingestion from uploaded .docx and .pdf files is supported through python-docx (≥1.1.0) and pypdf (≥4.0.0). External API

communication, including LinkedIn-style job data acquisition via a commercial provider, is implemented using httpx for asynchronous HTTP requests.

Configuration parameters (e.g., database URL, backend host/port, frontend origin, API keys) are managed through environment variables and optional .env files. The SQLite schema is automatically initialized upon application startup if not already present.

4.2.2. Frontend Implementation

The frontend is developed using React 18.2 with TypeScript 5.x, built and served via Vite 5.x. The user interface is implemented using Material UI (MUI) 5.x, and dashboard visualizations are rendered using Recharts 3.x. API communication and client-side caching are managed through TanStack React Query 5.x, while routing is handled using React Router DOM 6.x. Development mode is executed via npm run dev (default port 5173), while production builds are generated using npm run build and may be served through any static web server or reverse proxy. No containerization or orchestration tools are required for the proof-of-concept implementation.

4.2.3. Deployment and Execution Model

The backend is launched using: uvicorn app.main:app --host 0.0.0.0 --port 8000. The frontend launched using: npm install && npm run dev. Table 1 provides a summary of the used technologies.

Table 1. Summary of the technologies used.

Component	Technology
Backend	Python 3.11, FastAPI, Uvicorn
Database	SQLite 3, SQLAlchemy 2
Validation & Config	Pydantic 2, Pydantic-Settings
NLP & Text Processing	NLTK, langdetect, lexicon-based extraction
Document Parsing	python-docx, pypdf
HTTP Client	httpx
Frontend	React 18, TypeScript 5, Vite 5
UI & Charts	MUI 5, Recharts 3
Client Data Layer	TanStack React Query 5, React Router 6

4.3. EXPERIMENTS

We tested our proposed framework on the aforementioned dataset, containing job posts from the labor market in Jordan in the last month interval, we analyzed the results provided in the dashboard. Fig. 2 , Fig. 3, and Fig. 4 illustrate the results.

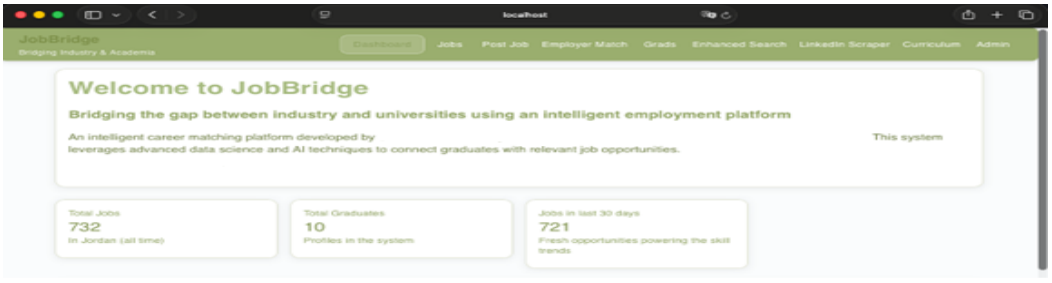

Fig. 2. Dashboard main page.

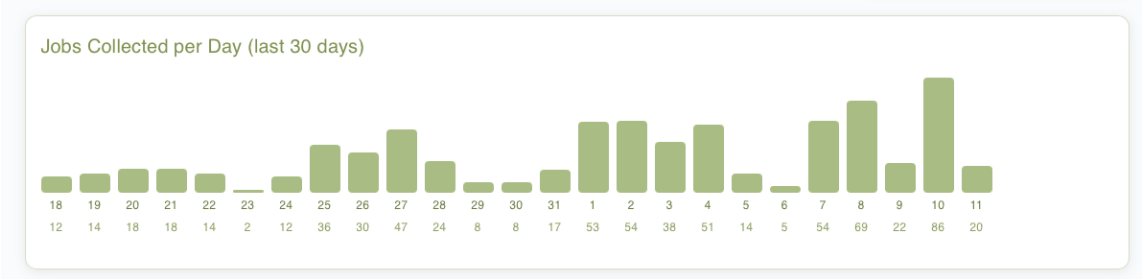

Fig. 3. Dashboard job statistics.

Fig. 4. The top 20 Extracted Skills.

The results in the dashboard indicate that the framework is able to scrap jobs, recognize skills, and extract trending skills. Given the scrapped jobs that represent job posts from the labor market in the last month interval, it is interesting to see that the framework extracted communication skills, leadership, problem solving, project management, and team work are among the top needed skills by the industry. Other reported skills were technical-based such as python, Azure, and SQL. We reported these results to two senior experts working in the industry for more than 10 years as software engineer and they confirmed that these results actually aligns strongly with the current trends in Jordan and the region. These results demonstrate the ability of our proposed framework to extract the up-to-date needed skills that can be used as a dynamic feedback to universities and individuals to make actionable decisions and continuous alignment.

Next, we studied in details the categories of the top 20 extracted skills from our proposed framework, and we compared them with the skills reported in national reports (DigiSkills, 2025). The results are provided in Table 2.

Table 2. Comparing extracted skills with skills reported in national skills.

Category	Skills Extracted by Framework	Skills Reported in National Reports	Match
Soft skills	Communication, Leadership, Teamwork	Communication, Leadership, Teamwork	Yes
Problem solving	Problem-solving	Problem-solving	Yes
Project management	Project management	Project management	Yes
Programming	Python, SQL	Python, SQL	Yes
Cloud	Azure	Cloud technologies	Yes
AI	AI	AI and automation	Yes
Software	Software engineering, Backend	Software engineering	Yes
DevOps	CI/CD	DevOps	Yes
Cybersecurity	-	Cybersecurity	No

From Table 2, we can easily find that the categories of the extracted skills by our framework matched the majority of the reported categories by national reports. In details, our proposed framework matched eight out of nine categories, which made the overlap 0.89. This demonstrates a strong alignment between the extracted skills and the Jordanian labor market needs. Furthermore, our proposed framework was able to extract not only technical skills such as programming languages but also competences such as soft skills, leadership, and teamwork. In addition, it extracted a third more abstract layer containing the domain name such software engineering and AI. This would enrich the required skills and competences and provide a more comprehensive feedback that can be used by many parties. The names of needed domains for example can be used by universities to open new programs in. Finally, our proposed framework, unlike many existing works, is dynamic and adaptive and can provide real time statistics on demand. The only skill category that was not extracted by our proposed framework despite its appearance in national reports is cybersecurity. However, this is not necessarily a weakness of the framework, as the extracted skills were based on the job posts In the last month only. Increasing the time interval to a year for example could eliminate such incidents. Next, we compared our extracted list with other extraction baselines. In details, we used both TF.IDF and LDA as extraction baselines. Table 3 provides the results.

Table 3. Comparing our proposed skill extraction framework with other baselines.

Method	P@5	R@5	F1@5	P@10	R@10	F1@10
Our Proposed Framework	0.98	0.42	0.59	0.97	0.83	0.9
TF.IDF	0.8	0.33	0.47	0.9	0.67	0.77
LDA	0.6	0.25	0.35	0.7	0.5	0.58

From Table 3, it is noted that our proposed framework outperformed other methods in both recall and precision. This proves that the framework can extract the correct skills and only the correct skills. Obviously, as the ground truth skills adopted from the national reports were more than 5 skills, recall@5 was low by design. However, it was much improved when using the top 10 extracted skills. Furthermore, precision was 1, which shows that when the model extract a skill, it is almost a correct one. TF.IDF was the second runner but it generated noisy and general terms. LDA was showed the worst performance, and this was expected as it is not specifically designed for skill extraction. Finally, we used the framework to extract skills on different time intervals. We extracted job posts from linkedin in Jordan labor market in the field of IT. The results were provided in Table 4.

Table 4. Comparing various extracted skills on different time intervals.

Extracted Skill	Score before one year	Score now	Grow Percentage
AI	12	24	100%
Cloud Computing(AZURE, AWS, …)	18	29	61%
DevOps/CI-CD	10	16	50%
Soft Skills	35	42	20%

From Table 4, it can be noted that the top growing demand was for the AI skills among the compared ones. This was expected as more and more companies and sectors are starting to use its components. Cloud skills are second runner as they are demanded more and more. It is interesting to see that some skills have higher demanding score but the overall growth is less such as soft skills. These skills have been requested by a wider range of job posts, and this explains the larger demand score. However, the growth of demand in the last year is smaller than other skills such as AI skills. These statistics can be fed into a machine learning model to predict demand and growth of skills in the coming years as well. This would further enrich the statistics and make them more comprehensive. This part is left for the future work. This can be further integrated with other educational components that benefited from artificial intelligence such as the extraction of student aptitudes based on their grades (Hawashin & Abusukhon, 2022). Such integration would complete and further enrich the educational process.

5. CONCLUSIONS

This work proposed a continuous data-driven framework to extract trending skills from online job posts, which would allow a real time alignment between universities and market needs. Unlike traditional periodic labor market previous studies, the proposed framework allows continuous and real-time monitoring of the demanded skills, allowing for a more adaptive and faster curriculum and workforce strategies. The benefits of this framework would include not only universities and industry but also accreditation processes, national digital transformation, and workforce development initiatives by providing a continuous data-driven approach. When compared against the national reports, our extracted skills showed a high alignment, outperforming TF.IDF and LDA baselines. Furthermore, our extracted skills cover not only technical terms but also competences. In addition, our framework can provide both demand score and growth percentage, which would indicate current and future demand for each skill. These skills can be fed periodically via a specific dashboard to universities to enable a dynamic curriculum update process. It can assist also in directing the training workshops conducted by universities toward the real demanded skills. Individuals and other involved sectors can also benefit from these skills. Our proposed framework would make a continuous approach to find current labor demands.

Future works can be done in various directions. The statistics obtained from the dashboard can be ingested in a machine learning model to provide further forecasting and pattern discovery. This would allow proactive skill forecasting and long-term workforce planning. Moreover, this framework can be scaled to include job posts from several platforms. This could include also regional and global markets. Finally, this work can benefit from the current development in transformers to further improve its accuracy and semantic understanding. This work represents a step toward building a more adaptive and dynamic education system that can adapt effectively to the rapidly changing developments.

ACKNOWLEDGEMENT: This work was funded by Al-Zaytoonah University of Jordan in the research project number 2025-2024/06/18. The authors would like to thank Al-Zaytoonah University of Jordan for its support.

REFERENCES

[1] I. Freitas, R. Marques, E. Silva, ''University–industry collaboration and innovation in emergent and mature industries in new industrialized countries,'' *Research Policy,* vol. 42, no. 2, pp. 443–453, 2013, doi: 10.1016/j.respol.2012.06.006.

[2] M. Plessis, N. Barkhuizen,N. Schutte, ''The impact of university–industry collaboration on graduate employability after COVID-19: A literature review,'' *Journal of Higher Education Policy and Management*, vol. 10, no. 6, pp. 30-41, doi: 10.18775/ijmsba.1849-5664-5419.2014.106.1003.

[3] M. Cohen, G. Fernandes, P. Godinho, ''Measuring the impacts of university–industry R&D collaborations: A systematic literature review,'' *Journal of Technology Transfer*, vol. 50, no. 1, pp. 345–374, 2025, doi: 10.1007/s10961-024-10114-5.

[4] J. Clemente-Mediavilla, R. Antolín-Prieto, ''LinkedIn job offers aimed at advertising graduates in Spain,'' *El Profesional de la Información*, vol. 28, no. 6, p. e280513, 2019, doi: 10.3145/epi.2019.nov.13.

[5] A. Almaleh, M. Aslam, K. Saeedi, N. Aljohani, ''Align my curriculum: a framework to bridge the gap between acquired university curriculum and required market skills,'' *Sustainability,* vol. 11, no. 9, p. 2607, 2019, doi: 10.3390/su11092607.

[6] K. Sjöö, T. Hellström, ''University–industry collaboration: a literature review and synthesis,'' *Industry and Higher Education*, vol. 33, no. 4, 2019, doi: 10.1177/0950422219829697.

[7] S. Ankrah, O. Al-Tabbaa, ''Universities–industry collaboration: a systematic review,'' *Scandinavian Journal of Management,* vol. 31, no. 3, pp. 387–408, 2015, doi: 10.1016/j.scaman.2015.02.003.

[8] B. Hawashin, A. Abusukhon, ''An efficient course recommender using deep-enriched hidden student aptitudes,'' *ICIC Express Letters, Part B: Applications,* vol. 13, no. 12, p. 1331, 2022, doi: 10.24507/icicelb.13.12.1331.

[9] R. Rybnicek, R. Königsgruber, ,'What makes industry–university collaboration succeed? A systematic review of the literature,'' *Journal of Business Economics,* vol. 89, no. 2, pp. 221–250, 2019, doi: 10.1007/s11573-018-0916-6.

[10] DigiSkills, *Jordan digital skills supply and demand gap analysis report*, Ministry of Digital Economy and Entrepreneurship, Government of Jordan, 2025.

[11] J. McLaughlin, C. Lupton-Smith, M. Wolcott, ''Text mining as a method for examining the alignment between educational outcomes and the workforce needs,'' *Education in the Health Professions,* vol. 1, no. 2, pp. 55–60, 2018, doi: 10.4103/EHP.EHP_25_18.

[12] D. Jackson, ''Employability skill development in work-integrated learning: Barriers and best practice,'' *Studies in Higher Education,* vol. 40, no. 2, pp. 350–367, 2015, doi: 10.1080/03075079.2013.842221.

[13] M. Perkmann, H. Schildt, ''Open data partnerships between firms and universities: The role of boundary organizations,'' *Research Policy,* vol. 44, no. 5, pp. 1133–1143, 2015, doi: 10.1016/j.respol.2014.12.006.

[14] P. Tambe, ''Big data investment, skills, and firm value,'' *Management Science,* vol. 60, no. 6, pp. 1452–1469, 2014, doi: 10.1287/mnsc.2014.1899.

[15] L. Bu, Y. Li, Z. Wang, ''Research on the impact of university–industry collaboration on green innovation: the mediating role of social capital and dynamic capabilities,'' *Sustainability*, vol. 17, no. 11, p. 5068, 2025, doi: 10.3390/su17115068.

[16] S. Khan, R. Ahmed, M. Malik, ''University–industry collaboration for academic success and employability: perspectives of students and practitioners,'' Studies in Higher Education, pp. 1-21, 2025, https://doi.org/10.1080/03075079.2025.2545606.