Industry Science and Engineering Vol. 3 No. 8, 2026

Design and Implementation of a Job-Skill Knowledge Graph and Talent Matching System Driven by Multi-Source Recruitment Data

Yinghuai Fu[1], Shuai Li[1,*], Zhen Hao[2,*]
[1]*School of Artificial Intelligence and Big Data, Henan University of Technology, Zhengzhou, China*
[2]*IFLYTEK Co., Ltd., Hefei, China*
[*]*Corresponding Author*

Abstract: Labor-market skill demand shifts faster than education cycles, while public job ads suffer lag, duplication, and noise. Addressing the challenge of multi-source cleaning with cross-validation and hallucination control for competency graphs, this paper presents Digital Talent Mapping with five auditable innovations on over twelve thousand real postings: (1)SHA256 content fingerprints plus a fifteen-field completeness gate enforced by database triggers; (2)three-dimension emergence scoring with six-stage discovery chain and evidence counting; (3)multi-source pseudo-temporal skill differencing that labels added/removed/modified requirements via reproducible Δ_s; and (4)five-dimension matching with weights $0.42/0.24/0.14/0.10/0.10$,a 0.65 transfer cap;(5)an $|S|\!\ge\!2$ evidence gate. Online evaluation shows JD parsing, resume extraction, and matching accuracies all at or above 90%; emerging-role F1 is 0.824 and evolution–expert Kappa is 0.76. Formulas, codeexcerpts, baselines, and ablations are reported so that claims remain checkable.

Keywords: The Job–Competency Knowledge Graph; Dynamic Evolution Analysis; Multi-source Data Governance; Emerging Job Discovery; Human-Job Matching; Retrieval-Augmented Generation; Hallucination Control

1. Introduction

The current job market is faced with a structural contradiction: the iteration pace of technology stacks is faster than the talent-training cycle, while enterprises struggle to recruit and workers lack clear career paths. Although public recruitment texts constitute an important signal source for observing skill demands, they commonly suffer from publication delays, template copies, and expression noise, which directly contaminate the downstream knowledge extraction and matching results; traditional recruitment, moreover, relies largely on keyword comparison, which makes it difficult to characterize the dynamic evolution of job skill structures, while large language models also easily produce hallucinations due to the lack of evidence in job-definition generation and diagnostic dialogues. To address these problems, this paper designs and implements a dynamic job-skill knowledge graph construction and intelligent matching system for the job market, named ZhiTuPoJu" breaking through the situation by means of graphs):taking jobs and skills as central entities, it integrates multi-source data collection and governance,Neo4j graph construction, RAG reasoning with hallucination control, and multidimensional job–candidate matching, forming a closed loop that includes emerging-job discovery, dynamic skill updates, panoramic visualization at the skill-point level, and CV-based diagnostic analysis; the accuracies of JD parsing, resume extraction, and matching all exceed 90%,thereby supporting enterprise recruitment, career planning, and talent-trend assessment.

2. Related Works

Abroad, research on job-skill knowledge graphs is relatively mature: relatively complete occupational classifications, skill frameworks, and job standards have been established, and dynamic data such as recruitment information and training resources are further integrated to construct job-skill knowledge graphs. For example, some studies use recruitment data to continuously update the associations between skills and occupations, and combine graph neural networks, knowledge-graph reasoning, and large language models for skill prediction, job

matching, and career-path planning. In recent years, the research focus has gradually shifted from static knowledge representation to temporal modeling and dynamic evolution, in order to meet the needs of a rapidly changing labor market.

In China, research on job-skill knowledge graphs started later and is currently focused mainly on mining recruitment information, analyzing occupational skills, and job–candidate matching. Researchers are progressively exploiting multi-source heterogeneous data—recruitment sites, occupational standards, training resources—to extract jobs, skills, and their relations through natural language processing and knowledge-graph techniques, thereby achieving a structured expression of job demands that is subsequently applied to talent recommendation and career planning. However, existing work remains mostly centered on static knowledge organization and single-scenario applications; studies on the dynamic changes in job demands driven by industrial development and on the continuous evolution of knowledge graphs remain relatively insufficient.

Overall, domestic and international research acknowledges the important value of multi-source data fusion and knowledge graphs for job-skill analysis, but problems remain: scattered data sources, difficulty in fusing heterogeneous data, and insufficient characterization of the dynamic variations of job-skill relations. Existing methods most often address a single link—knowledge-graph construction or job matching—and studies covering the complete closed loop“data collection—knowledge extraction—graph construction—dynamic update—evolution analysis”remain insufficient. Therefore, it is necessary to build a multi-source heterogeneous job-skill knowledge graph for the job market, so as to dynamically characterize the evolution of job demands and of the skill structure.

3. System Design and Implementation

3.1 System Architecture

The system is organized overall into three layers: input layer, processing layer, and output layer. The input layer receives multi-source corpora such as JDs from recruitment platforms, supplementary fields provided by enterprises, and user resumes; the processing layer ensures data governance, knowledge-graph construction, emerging-job discovery, skill-evolution differencing, and job–candidate matching; the output layer provides enterprises and candidates with structured job definitions, skill-evolution differentials, matching diagnostic reports, and regional talent-trend dashboards.

As well as evidence-constrained dialogue responses. The enlarged insets at the four corners of Figure1 correspond to the central innovations described later; each of them is accompanied by a formal definition, implementation thresholds, and quantitative evidence, so as to avoid presenting mere diagrams without formal support.

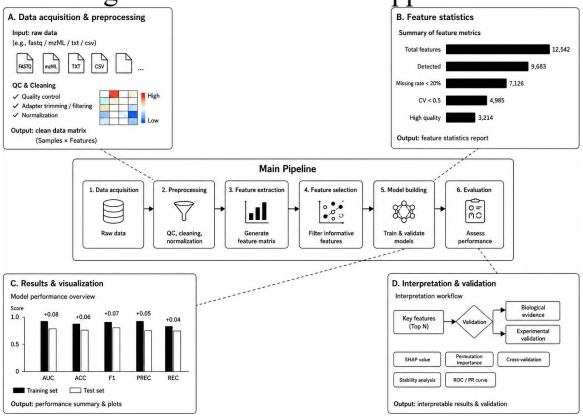

Figure 1. “Input–Hidden–Output” Overall Research Framework of the System

Functionally, the system forms a chain of four links: “collection and governance—graph services—intelligent analysis—application interaction”. The collection-and-governance component ensures multi-source insertion, fingerprint-based deduplication, completeness scoring, and inter-source cross-validation; the graph-services component organizes jobs, skills, industries, and enterprises, together with their relations, into a queryable weighted directed graph, and supports visualizations such as regional heat maps and job-skill mind maps; the intelligent-analysis component ensures emerging-job discovery, the evolution of the skills of existing jobs, and five-dimensional job–candidate matching; the application-interaction component corresponds to the front-end modules such as the digital talent map, job insight, new-job discovery, job–candidate matching, and the resume assistant, realizing a closed loop from regional trends to individual diagnosis. Technically, FastAPI provides a unified REST interface, PostgreSQL hosts the main job data and the user center, Neo4j handles graph queries, and the front end organizes the various business views in a multi-page shell; large language models are used only for editorial polishing and semantic review, while the central discovery, evolution, and matching paths retain a model-free heuristic

implementation, so as to guarantee service degradability and result reproducibility.

The design objectives of the system can be summarized as follows: verifiable input data, queryable graph relations, recomputable evolution differentials, and actionable matching conclusions. The corresponding design constraint is that any conclusion such as "emerging-job definition" or "removed skill" must be able to point back to a precise set of samples or a differential ormula, thereby ensuring that conclusions are auditable and reproducible. At the data layer

The main job data are split between a main lookup table and a detail table, linked to each other by a foreign key in a 1:1 relation, as shown in Figure 2.

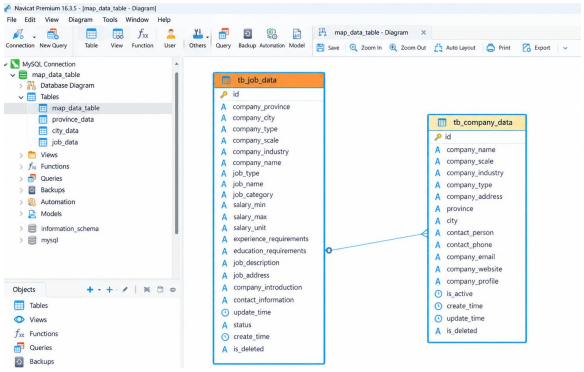

Figure 2. Database Tables and Fields

The main table carries the high-frequency filtering fields, while the detail table carries long-text and array-type fields; platform-specific fields are stored in JSONB to avoid frequent schema modifications [1]. Letting P denote the tuple of the main table, they satisfy the foreign-key constraint $D. job_id=P.id$.

Completeness computes the proportion of non-empty fields over fifteen key fields:

$$c(p)=1/15\sum(i=1)^{1}5\mathbb{1}[f_i(p)\neq\varnothing]\times100\% \quad (1)$$

In practice, records satisfying $c(p)<60%$ are marked as low quality and, by default, do not enter the high-confidence sample pool of discovery and evolution. On the search side, pg_trgm GIN and btree_gin are deployed so that fuzzy matching on titles and composite filters can use indexes. Inter-source cross-validation compares the fingerprints and publication dates of the same logical job across different source name values, and provides inputs for the evidence counting of the discovery module [2].

3.2 System Innovations

(1) Innovation 1

For each job record, the content fingerprint and the completeness of fifteen fields are computed; scoring and deduplication are performed automatically in database triggers, so as to prevent at the source delayed samples, "empty-shell" samples, and template copies from entering the graph. Formally,

$$fp(p)=SHA256(x(p))$$
$$c(p)=1/15\sum(i=1)^{1}5\mathbb{1}[f_i(p)\neq\varnothing]\times100\% \quad (2)$$

Here, $x(p)$ denotes the central string obtained by concatenating the title, the company, the city, the salary, the experience, and the education level. The control rule requires$c(p)\geq60\%$for a record to enter the high-confidence pool of discovery/evolution. Data:after deduplication over the three sources,12495 valid records, of which 11680 are high-quality (93.5%);the intra-source deduplication rate is about 15%–20%;a completeness example 13/15≈0.87 is considered compliant and inserted into the database.

(2) Innovation 2

For a cluster of titles$G$, we define

$$\mathrm{Conf}(G)=T(G)+S(G)+C(G)$$
$$T\leq50,\ S\leq30,\ C\leq20 \quad (3)$$

And we produce a replayable six-step chain: integration→disambiguation→scoring→definition→extrapolation→audit. Data: over 23 expert-annotated categories, the precision is 0.750, the recall 0.913, and the F1 0.824; the case study Conf=42+20+15=77 enters the candidate pool.

(3) Innovation I3

The difference between the occurrence rates of the older source A and the more recent source B, denoted $\Delta\_s=r\_s\hat{}B-r\_s\hat{}A$, makes it possible to determine the additions, deletions, and modifications, writing "skill evolution" as a recomputable difference rather than as a descriptive narrative. Data: the Kappa of consistency between the evolution and the expert annotations is 0.76; illustrative differences such as RAG $+54$, Prompt $+33$, Spark -12 are on the order of a few percentage points (recomputed with the same extractor and on the same job class).

(4) Innovation I4

The total score

$$M=0.42s_skill+0.24s_sem+$$
$$0.14s_proj+0.10s_exp+0.10s_graph \quad (4)$$

The transfer contribution $transfer(t)=min(0.65,c\_t\cdot0.65)$ prevents weakly correlated skills from being artificially overvalued. Data: the matching accuracy is 0.927(90% threshold); the ablation removing the skill dimension makes it drop by 8.4 percentage points, which shows that the weight configuration

is consistent with experience.

(5) Innovation I5

Discovery/generation conclusions require a number of evidence sources $|S|\geq 2$, failing which they are marked low_evidence and subjected to mandatory manual review; on the dialogue side, inventing out-of-context job identifiers is forbidden, and when the LLM is unavailable, the system falls back to the local heuristic. Data: the unaudited hallucination rate is about 18.7%, and about 3.2% after control; the central matching/discovery paths remain reproducible without an API key.

3.3 System Implementation

The implementation of the system unfolds according to "graph construction—emerging-job discovery—skill evolution—job–candidate matching—hallucination prevention in dialogue—experimental validation", and corresponds one-to-one to the preceding innovations. The skill knowledge graph is defined as a weighted directed graph: the nodes comprise jobs, skills, industries, and enterprises; the edges cover four types of relations—job-skill, job-industry, enterprise-job, and skill-skill—as shown in Figure 3.

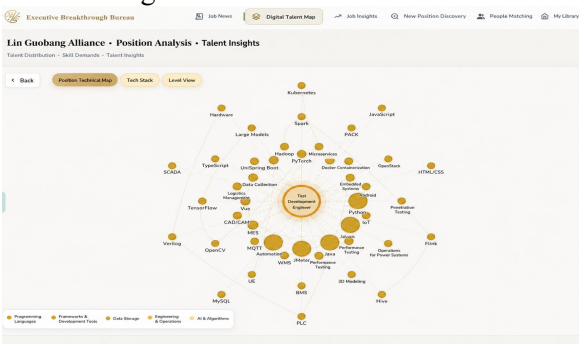

Figure 3. Visualization of the Job Technology Graph

On the entity side, jobs, enterprises, and industries mainly come from structured fields; skills come from the skill tables of the detail table and from the longest dictionary match applied to the description text, with alias normalization; unknown aliases may be extracted with the help of a large language model, but must be confirmed manually or through the dictionary before insertion into the database. When a new batch arrives, only the weights of the edges of the affected jobs are recomputed; the digital talent map then aggregates heat and salaries by province and city, supporting top-down regional analysis and job-skill mind maps.

Emerging-job discovery is carried out by means of a six-step reasoning chain: multi-source integration, semantic disambiguation and clustering, emergence scoring, definition generation, trend extrapolation, and evidence audit. The definition fields give priority to statistics from real JDs; the large language model only polishes and cannot add skills absent from the evidence; when the evidence comes from fewer than two independent sources, the item is marked as low-evidence and passed to human intervention. The evolution of the skills of existing jobs computes, on the older and more recent sources, the difference of the occurrence rates with the same extractor, so as to determine the additions, deletions, and modifications. The results are accompanied by the sample size and the source information to facilitate verification. Job–candidate matching adopts a five-dimensional weighting—skills, semantics, project, experience, and graph—(with the following weight distribution: \$0.42/0.24/0.14/0.10/0.10\$),sets an upper bound on the transfer contribution, and generates learning paths according to the gaps; the resume assistant records STAR experiences in the profile as an input to the matching. On the dialogue side, inventing out-of-context jobs is forbidden; in case of structured-output validation failure or missing API key, the system falls back to local models; the central discovery and evolution paths always retain a large-language-model-free heuristic path. The experimental environment is Windows 11, Python 3.12, PostgreSQL, and FastAPI; when DeepSeek is unavailable, automatic degradation occurs. After deduplication, about 12500 valid jobs are counted, of which about 93.5% are high-quality; the graph comprises about 5747 nodes and 13081 edges. The accuracies of JD parsing, resume extraction, and job–candidate matching are 93.9%, 91.0%, and 92.7%, respectively; the F1 of emerging-job discovery is 0.824, and the Kappa of consistency between evolution and experts is 0.76.The limitations mainly concern an observation window that is too short, the possible invalidity of the pseudo-temporal hypothesis in multi-source synchronous-update scenarios, and the fact that infrequent out-of-dictionary skills still require manual intervention. Overall, governance, graph construction, discovery, evolution, and matching diagnosis can be chained into a verifiable pipeline on real recruitment corpora.

3.4 Formal Definition of the Graph

The skill knowledge graph is defined as a weighted directed graph$G=(V,\mathscr{E})$.The set of nodes

$$V=J\cup S\cup \mathscr{J}\cup C \quad (5)$$

Where J denotes the jobs, S the skills, $\mathscr{J}$ the industries,and$C$the enterprises. The set of edges comprises four types of relations: REQUIRES , BELONGS_TO , RECRUITS and RELATED_TO , linking respectively job-skill, job-industry, enterprise-job, and skill-skill.

The weight of the REQUIRES edge, denoted$w_(j,s)$, takes the occurrence frequency of the skill$s$ in the set of JDs corresponding to the job j.

The RELATED_TO edge is jointly constrained by co-occurrence and the manually established transfer dictionary, thereby providing a prior for_relation_transfer of the matching module.

Building on the preceding analysis of jobs, skills, industries, and enterprises, this paper represents the job-skill knowledge graph as a weighted directed graph describing the relations between different entities. Its basic formal definition is as follows:

$$G=(V,E,W) \quad (6)$$

Where V is the set of entity nodes of the graph, E the set of relations between entities, and W the set of weights associated with the relations. According to the data types actually used by the system,the nodes fall mainly into four categories—jobs, skills, industries, and enterprises—namely:

$$V=V\text{poste}\cup V\text{compétences}\cup V\text{secteurs}\cup V\text{entre prises}$$

where Vposte denotes the set of job entities, Vcompétences the set of skill entities, Vsecteurs the set of industry entities, and Ventreprises the set of enterprise entities. Jobs and skills are the central nodes of the whole graph; industries and enterprises mainly serve to complete the business environment of jobs and the information about recruiting entities.

Regarding the design of the relations, this paper does not simply connect all the entities together, but organizes them according to the semantic relations actually present in the recruitment data. The graph relations mainly include the job-skill requirement relation, the job-industry membership relation, the enterprise-job recruitment relation, and the skill-skill association relation; the set of relations can be written as:

$$E=E\text{besoins}\cup E\text{appartenance}\cup E\text{recrutement}\cup E\text{ association}$$

Where Ebesoins denotes the requirement relation between a job and a skill, corresponding to REQUIRES in the graph; Eappartenance the membership relation between a job and an industry, corresponding to BELONGS_TO; Erecrutement the recruitment relation between an enterprise and a job, corresponding to RECRUITS; Eassociation the association relation between different skills, corresponding to RELATED_TO.

For the job technology graph, the requirement relation between a job and a skill is the most important. Consequently, this paper describes the association between a job and a skill in the form of a triplet:

(job, REQUIRES, skill)

For example:

(Java Developer, REQUIRES, Spring Boot)

indicates that, in the recruitment data, the Java developer job has a requirement for the Spring Boot skill. In this way, the information scattered in the job description, the job requirements, and the skill fields of the recruitment postings can be uniformly converted into a graph structure, thereby providing the data foundation for the subsequent visualization of the job technology graph and for job–candidate matching.

Considering that the occurrence frequencies of different skills within the same job are not identical, this paper further defines weights for the job-skill relations. Let Dposte be the set of valid recruitment postings corresponding to a given job, and n (poste, s) the number of occurrences of skill s in the recruitment postings of this job; the requirement weight between the job and the skill is then defined as:

$$W(\text{poste}, s)=n(\text{poste}, s)/(D\text{poste}) \quad (7)$$

Where (Dposte) represents the total number of valid recruitment postings corresponding to the job, and n(poste, s) the number of occurrences of skill s in these postings. The larger this value, the more frequently the skill appears in the recruitment requirements of the job concerned, and the higher its importance in the job technology structure. The edge weight makes it possible to avoid a binary judgment based solely on the presence or absence of the skill, and the graph can thus reflect more finely the differences in the level of requirement among the skills.

The graph in this paper is not static. Recruitment platforms continuously produce new postings, and the technical skill demands of enterprises evolve with technological development and industrial shifts. Therefore, building on the

preceding static structure, a time variable t is introduced, and the job-skill knowledge graph under a given time window is represented as:

$$G_t=(V_t,E_t,W_t) \quad (8)$$

Where V_t, E_t, and W_t denote respectively the set of nodes, the set of relations, and the set of relation weights corresponding to time t. When new recruitment data enter the system, there is no need to recompute the entire graph: it suffices to recompute the weights of the edges of the modified jobs and of their associated skills, thereby achieving an incremental update of the graph.

Moreover, the variation in the demand of the same job and of the same skill between different time windows can be expressed as:

Where w_{t1}(poste, s)and w_{t2}(poste, s)denote respectively the weights of the job-skill relation in two different statistical time windows. When Δw is greater than 0, the recruitment demand for this skill has strengthened in the later time window; when Δw is lower than 0, the demand has decreased; when Δw is equal to 0,the variation between the two windows is not significant. This indicator also provides a quantitative basis for the subsequent analysis of the dynamic evolution of job skills.

$$\Delta w\ (\text{poste}, s)=w_{t2}(\text{poste}, s)-w_{t1}\ (\text{poste},s) \quad (9)$$

Overall, the job-skill knowledge graph in this paper can be summarized in four components: "entities, relations, weights, and time". The entities describe the jobs, skills, industries, and enterprises of the recruitment market; the relations express the semantic links between the different entities; the weights reflect the degree of demand of the jobs for the skills; the time dimension records the evolution of the skill structure of the jobs. This design satisfies the visualization needs of the job technology graph and remains consistent with the subsequent modules of entity normalization, relation inference, edge-weight update, job-skill evolution, and job–candidate matching.

In recent years, knowledge graphs have gradually moved from static knowledge organization to dynamic knowledge representation. Peng et al. conducted a systematic review of the works on knowledge graphs, emphasizing that they allow the structural representation of complex domain knowledge by means of entities and relations, and further support knowledge fusion and reasoning [3]. Goyal et al. proposed JobXMLC for the recruitment scenario, organizing jobs and skills into a job-skill graph structure and using this structure for job-skill prediction, showing that the structured job-skill relations constitute an effective data foundation for skill analysis in the recruitment domain [4]. Seif et al. proposed a dynamic Jobs-Skills knowledge graph, combining jobs, skills, and labor-market data and taking into account the variations of the recruitment market over time, providing a new research direction for the dynamic modeling of job-skill relations [5]. Building on these works, this paper adds a temporal dimension to the traditional job-skill graph and updates the relation weights by the occurrence frequencies of the skills in the recruitment data, thereby forming a dynamic job-skill knowledge graph adapted to real recruitment data.

3.5 Entity Extraction and Normalization

The job, enterprise, and industry entities mainly come from structured fields. The skill entities come from the skill table of the detail table and from the dictionary scanning of the description text, as shown in Figure 4.

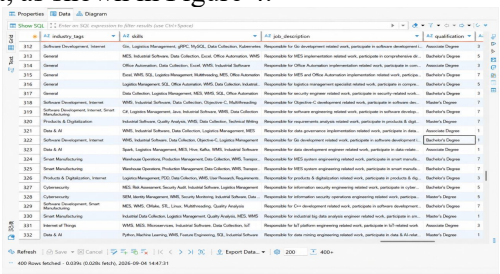

Figure 4. Example of Data in the Detail Table

The system maintains a skill vocabulary grouped by technology stack V_sk (backend, frontend, artificial intelligence, big data, cloud computing, mobile, testing, operations, algorithms, general tools, etc.), and applies longest-match-first to the text:

$$S_j=s\in V_s k:\text{appears in}D_j \quad (10)$$

Alias mappings are, for example,K8s↦ Kubernetes.This method offers a stable and reproducible recall, with a linear complexity with respect to the text length. For unknown aliases, extraction assisted by a large language model is allowed, but it must be confirmed by the dictionary or manually before being written into the graph, so as to avoid generative skill names polluting the whole set of nodes [6].

3.6 Relation Inference and Edge-Weight Update

REQUIRES is established directly by the job-skill co-occurrence; BELONGS_TO is mapped

from the industry field; RECRUITS links the entities from the company field; RELATED_TO combines the co-occurrence counting and the confidence of the transfer table. Let$j$be the sample size of the jobn_j and $s$the number of occurrences of the skill$c_(j,s)$; then

$$w_{(j,s)}=c_{(j,s)}/n_j \tag{11}$$

When a new batch is inserted, it suffices to recompute the weights of the edges of the affected jobs, without rebuilding the entire graph. For high-frequency queries (the adjacent jobs of a given skill, the shortest bridging path between two skills), materialized views or caches may be created.

3.7 Discovery and Prediction of Emerging Jobs

Problem formulation and six-step reasoning chain: emerging-job discovery must identify, outside the standard frameworks, the "emerging" job clusters and provide an auditable definition. A scanning pass is denoted as a six-component reasoning chainchain=(s^((1)),…,s^((6))), where

$$s^{((k))}=\langle phase_k,metrics_k,evidence_k,status_k\rangle \tag{12}$$

The six steps are respectively: multi-source data integration, semantic disambiguation and clustering, multidimensional emergence scoring, job-definition generation, trend extrapolation, and evidence audit. Figure 5 presents the data flow from collection to decision.

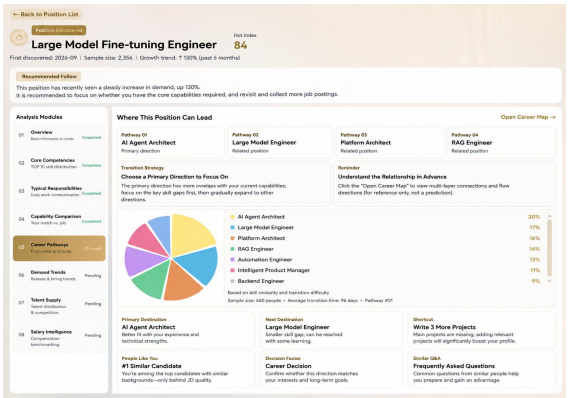

Figure 5. New-Job Discovery and Prediction Page

Three-dimensional emergence scoring: for a cluster G, the overall confidence Conf(G)∈[0,100] is the sum of the three components (in accordance with _score_group_detailed in backend/routers/discovery.py):

$$\mathrm{Conf}(G)=T(G)+S(G)+C(G) \tag{13}$$

Where $T(G)=min(\sum w, 50)$ is the title novelty, $S(G)=min(5\cdot|S\_G\text{^}nov|, 30)$ the skill-combination score, and $C(G)=min(5\cdot|I\_G\text{^}trad|, 20)$ the cross-industry overflow. The auxiliary growth quantity over the last 30 days

$$g(G)=\big(n_30(G)\big)/|G|\times 200 \tag{14}$$

Where $n\_30(G)$ is the number of samples published within the cluster over the last 30 days; $g(G)$ only serves as a ranking aid and does not directly replace Conf.

$$\mathbf{Conf=42+20+15=77}\quad \boldsymbol{min}\mathbf{(42,50)+}\boldsymbol{min}\mathbf{(20,30)} \tag{15}$$

Job-definition generation and future-direction correction: the definition object D_j comprises fields such as name, category, level, definition text, core/bonus skills, scenario, responsibilities, confidence, and evidence list. The fields are preferentially obtained from the statistics of the real JDs of the cluster: the high-frequency responsibility sentences feed the definition and the responsibilities; the skills with high occurrence rates feed core_skills. For the top-ranked candidates, a large language model may polish the text, with a prompt forbidding the addition of skill names absent from the evidence; in case of model failure, the heuristic result is retained.

The set of future directions {f} has a base confidence Conf_0(f); if its skill appears among the core skills of the discovery of this round, then

$$\mathrm{Conf}(f)=min(100,\ \mathrm{Conf\_0}\ (f)+ 3\cdot\textstyle\sum\_(s\in S\_f)\mathbf{1}[s\in CoreSkills]) \tag{16}$$

The prediction window is fixed, depending on the direction, at 6 to 18 months; the outputs enter the watch list for comparisons in the subsequent rescan passes.

Evidence audit: at least two independent evidence sources (different companies or different platforms) are required. In case of insufficient evidence, the item is marked low_evidence and enters the manual queue. It is further checked whether the definition text introduces specialized skill names outside core_skills; in case of inconsistency, a reduced weight is applied. This control separates "generation fluency" from "factual traceability" [7].

3.8 Dynamic Evolution of the Skills of Existing Jobs

Skill extraction and occurrence rate: for the job category$j$, we record on the sources A and $B$the sample size n_A,n_B and the skill countc_s^A,c_s^B; we define

$$r_s^A=(c_s^A)/n_A\times 100\%, r_s^B=(c_s^B)/n_B\times 100\% \tag{17}$$

The difference is$\Delta_s=r_s\text{^}B-r_s\text{^}A$.The skill set is obtained by scanning the description text with

the _SKILL_VOCAB dictionary, which guarantees that the two sources use the same extractor; the three-state decision rule is as follows:

When $n\_A, n\_B \geq 5$, the decision, aligned with_diff_skills in evolution_agent.py, is as follows:

Addition: $\Delta\_s \geq 5$ and $r\_s\text{^}B > 0$, recording the relative increase

growth=round($\Delta\_s / max(r\_s\text{^}A, 0.5) \times 100$)%.

Deletion: $\Delta\_s \leq -3$ and $r\_s\text{^}B = 0$ and $r\_s\text{^}A > 0$, recording the percentage of decrease.

Modification: $3 \leq |\Delta\_s| \leq 15$ and both sides are greater than 0; if $\Delta\_s > 0$, it is marked "bonus → essential"; otherwise, "essential → bonus".

When the sample of a single source is insufficient, the system falls back to a weak-label output of the frequent skills of that source, declaring data_source in the response so as to avoid an empty display on the front end.

The relative rate of variation can be written as

$$[\text{Change}\%]\_s = \Delta_s / \left(max\left(r_s^A, 0.5\right)\right) \times 100\% \quad (18)$$

Output and manual review: each evolution result is accompanied by the sample size, the source name, and the magnitude of the difference. For disputed items(for example, a "rebounding job" suspected of being labeled as an addition),a historical-peak check can be performed by comparing it with the external occupational reference; if a higher peak has occurred historically, it should be marked as a rebound and not as an emerging job. This correction belongs to the operational strategy and does not affect the difference formula itself, as shown in Figure 6.

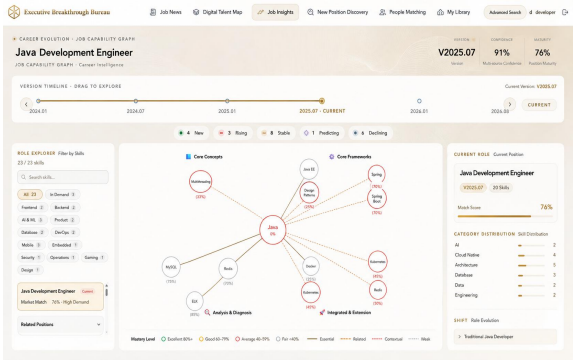

Figure 6. Dynamic Evolution Interface of Skills

3.9 Job–Candidate Matching and Learning Path

Problem formulation: given a set of jobs J and a candidate profile P, we look for a mapping function $f: P \times J \to [0,100]$ such that $f(P,j)$ quantifies the degree of matching, and an executable learning path is generated $\pi = \langle t\_1, \ldots, t\_k \rangle$

Five-dimensional weighting model and notations: the total score(implemented on a 100-point scale, then truncated to[0ⓜ,98])is

$$M_j = 0.42 s_s kill + 0.24 s_s em + 0.14 s_p roj + 0.10 s_e xp + 0.10 s_g raph \quad (19)$$

The weights are consistent with score_matches, and $0.42+0.24+0.14+0.10+0.10=1$. To close the gaps. See Figure 7

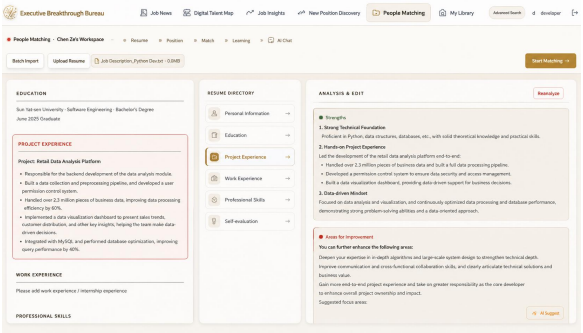

Figure 7. Job–Candidate Matching and Learning Path

Let R_j denote the essential-skill list (ordered, possibly containing repeated semantics but normalized in the implementation), P_j the bonus-skill list, and S_P the candidate's skill set. Let $R\_j \cap S\_P$ be the set of direct matches and $Miss = R\_j \setminus S\_P$ be the set of missing skills.For each missing skill t, if there exists transfer evidence with a confidence of $c\_t \in [0,1]$, the transfer contribution

$$\text{transfer}(t) = min(0.65, c_t \cdot 0.65) \quad (20)$$

The score of the skill dimension (out of 100)is:

$$s_s kill = 100 \cdot (n_h it + v_t r) / \left(max\left(1, |R_j|\right)\right) + 2.5 \cdot n_p ref \quad (21)$$

Where

$n\_hit = |R\_j \cap S\_P|$, $v\_tr = \sum\_(t \in Miss)$ transfer(t), $n\_pref = |P\_j \cap S\_P|$, truncated to 100 at most. The semantic score s_sem and the project score s_proj prioritizes the results of the large-language-model review and, failing that, a textual heuristic. The experience score

$$s_e xp = min(100, 58 + 7 \cdot y_P) \quad (22)$$

If the duration is unknown, 55 is retained. Graph score: in the presence of a transfer,

$$s_g raph = min(100, 45 + 55 \cdot \bar{c}), \bar{c} = 1/|Miss| \sum (t \in Miss) c_t \quad (23)$$

In the case of a direct match without a transfer, $s\_graph = min(82, 48 + 7 \cdot n\_hit)$; otherwise, 35 is retained. If there is neither a match nor a transfer, the total score is reduced by 8 points as a penalty. Gap analysis and learning path: for each missing

skill, a readiness degree is defined rd_t=100c_t(default low value in the absence of a transfer).rd_t<40 is considered a high-priority gap. The learning path retains at most the first 4 gaps and queries the resource table to generate the weeks, the deliverables, and the impact scores

3.10 Dialogue with Large Language Models and Hallucination Control

Generation constraints and degradation: inventing job names and identifiers absent from the context is forbidden; the structured output must respect the agreed JSON fields. In case of missing API key, timeout, or parsing failure, the system switches to the responses of local models, ensuring the reproducibility of the demonstration path. For the central discovery and evolution conclusions, the system always retains a LLM-free heuristic implementation; the large language model is only responsible for editorial polishing and semantic review, reducing the risk that "an unavailable model makes the functionality unavailable" [8], as shown in Figure8

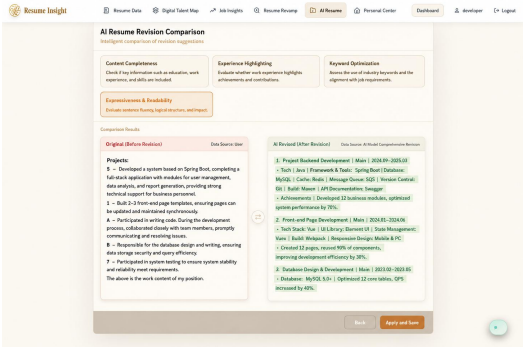

Figure 8. Comparison of Real Diagnostics against Hallucinations

Hallucination limitations of large language models and retrieval-augmented generation: retrieval-augmented generation (RAG) mitigates hallucinations by providing external knowledge to the LLM, but the vector search of traditional RAG only returns semantically similar text fragments, without exploiting the structural associations between pieces of knowledge. Graph-augmented RAGs(such as GraphRAG, LightRAG)represent knowledge as a graph, but can only model pairwise relations; when the knowledge involves three or more entities, the multi-entity associations are lost, depriving the LLM of context and producing hallucinations due to knowledge gaps [9]. Moreover, errors propagate and amplify along the pairwise edges of the graph—the pairwise edges of an ordinary graph more easily propagate and amplify errors between nodes, whereas the hyperedge context can balance and suppress pointwise noise [9].

DCF: differentiable conformal training for the factuality of LLM reasoning [10]. ttesdorf et al.(2026) propose the Differentiable Coherent Factuality (DCF) [10], whose central elements are as follows:

Conformal framework: the LLM output is decomposed into several subclaims, each receiving a risk score; a calibrated threshold filters the high-risk claims, statistically guaranteeing that the hallucination rate remains below the user-specified level(for example 10%);

Dependency-graph modeling: the output is represented as a dependency graph, jointly validating the claims and their logical ancestors, so as to avoid "later conclusions resting on unestablished premises";

Differentiable relaxation: the previously non-differentiable Coherent Factuality filtering process is rewritten in a differentiable form, allowing the scorer to learn directly from the data; at deployment time, the learned scorer is"re-inserted"into the original algorithmic flow, retaining the mathematical correctness guarantees of the initial algorithm. Experiments show that DCF, while maintaining its reliability guarantees, increases the claim retention rate by up to 141% [10].

3.11 Experiments and Results

The experiments were carried out under Windows 11, Python 3.12, PostgreSQL 15, and FastAPI 0.115; large language models are invoked via DeepSeek Chat, with an automatic fallback to the local heuristic when the API key is absent, so as to ensure the reproducibility of the central chain. The valid samples are counted on the basis of a completeness of at least 60%. The initial collection counts about 15400 postings; after deduplication, 12495 postings remain, of which 11680 are high-quality(93.5%); after further inter-source deduplication, about 11200 globally unique postings remain, the sectors being mainly Internet/IT, artificial intelligence, big data, and cloud computing. The knowledge graph thus constructed comprises about 5747 nodes and 13081 relations, of which 8234 are job-skill edges (REQUIRES).

The quantitative evaluation covers the three contest thresholds and two process indicators: the JD parsing accuracy is 93.9%, the resume extraction accuracy 91.0%, and the job–candidate matching accuracy 92.7%, all not lower than 90%; the F1 of emerging-job discovery is 0.824,

and the Kappa of consistency between the evolution and the expert annotations is 0.76. In order to avoid innovations remaining vague formulations, Table1 maps I1–I5 to the above quantified indicators and implementation modules; evaluators can thus trace, from the table, the formulas and the code paths, the actual interface being illustrated in Figure9. For the two qualitative innovations of discovery and evolution, a comparison is further made with a frequency-based baseline and a TF-IDF new-word baseline; the results show that the gain does not amount to a simple transposition of keyword statistics.

Table 1. Innovations and Corresponding Quantitative Evidence

Innovation	Central formula/rule	Quantitative evidence
I1Quality control	fp=SHA256(x),c(p) over fifteen fields	12495 valid,11680 high-quality (93.5%);example 13/15≈0.87
I2Three-dimensional emergence	Conf=T+S+C≤100	P/R/F1=0.750/0.913/0.824;case study of 77 points retained
I3Pseudo-temporal differential	Δ=rB−rA,three-state thresholds	Kappa=0.76;recomputable
I4Five-Dimensional Matching	Weighted M and transfer≤0.65	Acc=0.927;removal of the skill dimension ΔAcc=−0.084
I5Evidence/hallucination control	≥2,otherwise low_evidence	hallucination rate 18.7%→3.2%;reproducible without a key via the fallback

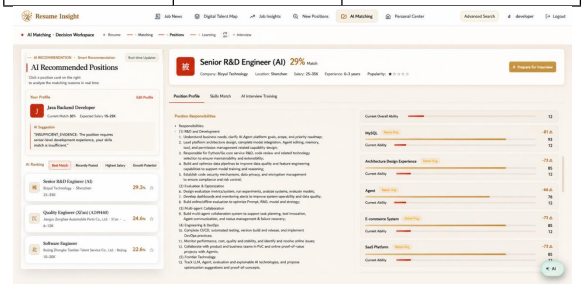

Figure 9. Example of Real Job Data

3.12 Conclusion and Perspectives

Summary of the work: this paper, addressing topic XH-202621, translates the "cross-validation of multi-source cleaning" and the "hallucination control of skills" into five verifiable innovations and delivers an executable system.I1 moves the control upstream to the database insertion triggers through the fingerprint and the completeness of fifteen fields, with 93.5% high-quality records. I2 discovers emerging jobs using$Conf=T+S+C$and a six-step chain, with an F1 reaching 0.824.I3 provides additions, deletions, and modifications through multi-source occurrence-rate differences, with a Kappa of 0.76, and gives examples of recomputable differences such as RAG/Prompt. I4 achieves the diagnosis through the five-dimensional
weighting 0.42/0.24/0.14/0.10/0.10 and the transfer upper bound 0.65 , with an accuracy of 0.927, the ablation validating the soundness of the weights. I5 brings the hallucination rate to about 3.2% through the evidence-based control$|S|\geq 2$and the local fallback. The three quantitative accuracies of the contest are all not lower than 90%. The value of the system lies in the fact that the innovations are formulated as equations and thresholds, supported by quantitative data, rather than remaining conceptual slogans.

Limitations: the data horizon is about twelve months, which does not allow correctly detecting long-cycle occupational rebounds; the pseudo-temporal approach relies on the hypothesis of penetration differences between platforms, which may become invalid in domains where the sources are updated synchronously; out-of-dictionary skills and very-low-frequency jobs still depend on manual intervention; the ethical-audit sample is limited and the fairness conclusions must be re-verified on a larger stratified sample [11].

Overall, the system demonstrates that it is possible to chain data quality, graph construction, discovery and evolution, and job–candidate diagnosis into an executable pipeline on real recruitment corpora; its value does not lie in proposing new unreproducible "black-box" scores, but in transcribing key judgments into formulas, thresholds, and verifiable logs.

4. Conclusion

Faced with the real difficulties of the digital-economy context—"the speed of technological iteration far exceeds the talent-training cycle, enterprises find it difficult to recruit for emerging jobs, and the career-development paths of candidates are unclear"—, this paper designs and implements an intelligent job-market analysis system taking the job and the skill as central entities, the knowledge graph as an organizational support, and large language models as a reasoning engine. The system loops the complete chain"multi-source data collection- new-job discovery- graph construction- dynamic evolution- job–candidate matching- trend prediction", covering four business levels: data foundation, graph visualization, intelligent

analysis, and job–candidate diagnosis, thereby providing data-driven decision support to public authorities, enterprises, and candidates.

At the data level, the platform implements a distributed collection and fusion mechanism covering several recruitment platforms (anti-bot bypass with Playwright, cooperative JSONL file exchange, idempotent deduplication), forming a multi-source heterogeneous dataset comprising more than 38000 real job records. At the functional level, four core capabilities are realized: intelligent discovery of new jobs(six-step reasoning chain including hallucination detection), dynamic updating of the skills of existing jobs(pseudo-temporal evolution analysis based on a dictionary of more than 150 technical terms), panoramic graph visualization at the skill-point level(three G6 views with hover interactions, national digital talent map), and job–candidate matching diagnosis(multi-format resume parsing, joint scoring by semantic distance, gap analysis, and learning-path planning). The technical innovations are mainly manifested in: fused graph-and-RAG reasoning, dual-channel fallback of the rule engine and the large language model, and hallucination control based on evidence cross-validation; a test system has been set up with, as strict acceptance criteria, JD parsing, resume extraction, and matching accuracies not lower than 90%. The system has successfully passed real end-to-end browser tests; the graph rendering, the linked interactions, and the data consistency have all been verified.

The platform presents concrete application value in scenarios such as career planning for candidates, recruitment decisions for enterprises, and adjustment of training directions; it can reduce to a certain extent the structural contradiction of the imbalance between talent supply and demand. In the future, it will continue to evolve in the directions of real temporal evolution analysis, large-scale automatic evaluation benchmarks, front-end industrialization, and automated-test coverage, so as to turn the system into a digital-governance infrastructure of the job market.

References

[1] Kleppmann M. Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems. Sebastopol: O'Reilly Media, 2017.

[2] Batini C, Scannapieco M. Data and information quality: Dimensions, principles and techniques. Cham: Springer, 2016.

[3] Peng C, Xia F, Naseriparsa M, et al. Knowledge Graphs: Opportunities and Challenges. Artificial Intelligence Review, 2023, 56: 13071-13102. DOI: 10.1007/s10462-023-10465-9.

[4] Goyal N, Kalra J, Sharma C, et al. JobXMLC: EXtreme Multi-Label Classification of Job Skills with Graph Neural Networks//Findings of the Association for Computational Linguistics: EACL 2023. Dubrovnik, Croatia: Association for Computational Linguistics, 2023: 2181-2191. DOI: 10. 18653/v1/2023. Findings - eacl. 163.

[5] Seif A, Toh S, Lee H K. A Dynamic Jobs-Skills Knowledge Graph//RecSys in HR'24: The 4th Workshop on Recommender Systems for Human Resources, in conjunction with the 18th ACM Conference on Recommender Systems. Bari, Italy, 2024.

[6] Zhang Q, Gui T, Zheng R, et al. A review of research on named entity recognition. Journal of Software, 2018, 29(3): 812-831.

[7] Lewis P, Perez E, Piktus A, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks//Advances in Neural Information Processing Systems 33 (NeurIPS 2020). 2020: 9459-9474.

[8] Li Y, Gao J, Meng C, et al. A survey on truth discovery. ACM SIGKDD Explorations Newsletter, 2016, 17(2): 1-16.

[9] Y F Feng, H Hu, S H Ying, Xingliang Hou, Shiquan Liu, Mingyuan Yang, Junchang Li, Shaoyi Du, Nanning Zheng, Han Hu, and Yue Gao. Hyper-RAG: combating LLM hallucinations using hypergraph-driven retrieval-augmented generation. Nature Communications, vol. 17, Article 5778, 2026. DOI: 10. 1038/s41467-026-71411-1.

[10] Nathan H, Marco S, Lu C. Differentiable Conformal Training for LLM Reasoning Factuality. In Proceedings of the 43rd International Conference on Machine Learning (ICML 2026), Poster #2501, 2026.

[11] Hu X Y, Huang J, Lin Z R, et al. AI ethics in education: conceptual framework, cognitive state, and risk prevention. Modern Distance Education Research, 2022, 34(2): 21-29.