A Trusted Computing Framework For Intelligent Employment Recommendation Of Industrial Talents Integrating Differential Privacy

Xiaojuan Liang, Chunjie Xie, and Wenting Qiu*

Minxi Vocational and Technical College, Longyan, 364000, China

Corresponding author. E-mail: fj139364000@163.com

Received: Jui. 20, 2026; Accepted: Aug. 13, 2026

Intelligent employment recommendation for industrial talent must process multi-source data — profiles, certificates, behavior logs, and company sources — yet existing methods emphasize accuracy over privacy and high-concurrency deployment. This study designs DP-TCERS, a differential-privacy trusted computing framework: it perturbs user/job features via local differential privacy and gradient noise against inference attacks, builds attention-fused towers for matching, scores trustworthiness from consistency and reliability, and lowers overhead via caching and parallel inference. Simulations show precision@10/recall@10/F1@10/NDCG@10 of 0.773/0.732/0.752/0.765 — below non-DP matching but above CF, NCF, DeepFM — with attack success rates falling to 0.198-0.231 and latency at 10,000 users reaching 294 ms versus 1,486 ms single-node.

Keywords: industrial talent recommendation, differential privacy, trusted computing, privacy-preserving recommendation, scalable information systems, and recommendation security.

http://dx.doi.org/10.6180/jase.202612_35.018

1. Introduction

Employment recommendation systems are key modules in industrial talent recruitment, online recruitment, and enterprise HR management. As employment data expands, manual screening and rule matching can no longer meet large-scale needs. Job seekers' background reflects employment ability, while job requirements reflect demand; recommendation must address cold start, heterogeneous data, and sparse feedback [1], with profile construction important for internship matching [2].

Algorithmically, job recommendation has evolved from content similarity and collaborative filtering to deep representation learning, with models jointly processing resume text, descriptions, and skill tags for stronger semantic expression than keyword matching [3], and LLMs further improving efficiency by extracting responsibilities and requirements [4]. Research has also shifted from single-accuracy to multi-objective optimization, where sources, feedback types, metrics, and deployment methods all affect effectiveness [5, 6].

Job recommendation differs from product recommendation since it directly affects careers and hiring decisions, requiring fairness, credibility, and interpretability beyond accuracy. Credibility covers accuracy, robustness, fairness, transparency, and privacy [7]; historical-interaction-only learning can reinforce popularity bias [8]; and explainable frameworks emphasize decision-consistent explanations [9]. This study positions job recommendation as a trustworthy, scalable information system.

Job recommendation systems handle highly sensitive data — education, skills, location, salary, and hiring preferences — that, if centrally trained, exposes attackers to member/attribute inference, back-inference, or gradient inversion. Glass-box modeling reduces data-abuse risk while increasing transparency [10]; federated and differential-privacy research reduces exposure risk [11]; and lightweight approaches like LightFR balance privacy

and efficiency [12].

Differential privacy offers a quantifiable framework: introducing random perturbations during queries, representation, or training so that adding or removing one user's data does not significantly change output. Privacy challenges mainly stem from client and gradient leakage, malicious aggregation, and inference attacks; local DP research shows Gaussian-mixture perturbations can protect rating data [13]; and homomorphic-encryption research shows protection needs both algorithm- and system-layer computation [14].

Beyond privacy, trusted computing is key: trusted job recommendation must explain suitability, verify source reliability, detect anomalies, and resist attacks. This study proposes DP-TCERS, integrating differential privacy to protect sensitive information while sustaining matching performance, low latency, and scalability under high-concurrency deployment.

The main contributions are: (1) a differential-privacy trusted computing framework unifying data representation, privacy-preserving recommendation, trustworthiness assessment, and scalable deployment; (2) dual feature/training-layer differential privacy reducing inference risks; (3) a dual-tower attention model improving stability under perturbation; (4) a trustworthiness score evaluating consistency, reliability, stability, and security; and (5) simulations verifying performance, privacy, attack resistance, latency, and scalability.

2. Methods

2.1. System Model and Problem Definition

2.1.1. *Intelligent Job Recommendation Scenario*

This study considers a job recommendation system consisting of job seekers, job resources, an employment platform, and enterprise recruitment nodes. Let the set of job seekers be

$$U = \{u_1, u_2, \ldots, u_m\} \tag{1}$$

The job postings are as follows.

$$J = \{j_1, j_2, \ldots, j_n\} \tag{2}$$

Where u_i represents the i job seeker, and j_k represents the k job posting. The goal of the job recommendation system is to generate a Top-K job recommendation list for each job seeker u_i, while protecting user privacy.

$$R_i = \{j_{i1}, j_{i2}, \ldots, j_{iK}\} \tag{3}$$

Recommendation lists must meet accurate matching, privacy/security, reliable sourcing, and efficient response, since profiles are highly sensitive, job data is dynamic, results carry decision-making impact, and platforms need scalable service for large user bases.

2.1.2. *Multi-Source Employment Data Representation*

A job seeker profile consists of basic information, skill characteristics, behavioral characteristics, and preference characteristics and can be represented as:

$$x_i = [s_i, e_i, c_i, b_i, p_i] \tag{4}$$

Where s_i represents professional and skill tags, e_i represents educational background, c_i represents certificates and ability levels, b_i represents job-seeking behaviors such as browsing, saving, applying, and interviewing, and p_i represents preferences for job location, industry, salary, and job type.

The job profile is represented as follows.

$$y_k = [r_k, q_k, l_k, w_k, d_k] \tag{5}$$

Where r_k represents job responsibilities, q_k represents job qualification requirements, l_k represents work location, w_k represents salary range, and d_k represents job description text. Job description text is largely unstructured, so this paper uses a text encoder to map it into semantic vectors.

User interactions with job postings include browsing, saving, applying, interviewing, and receiving hiring feedback. Let the interaction matrix be

$$A = \{a_{ik}\}_{m \times n} \tag{6}$$

Here, $a_{ik} = 1$ indicates that user u_i has a positive interaction with job j_k, and $a_{ik} = 0$ indicates that there is no explicit positive interaction. To accommodate the prevalence of implicit feedback in job recommendations, this paper assigns different weights to browsing, saving, applying, and interviewing:

$$a_{ik} = \omega_1 v_{ik} + \omega_2 f_{ik} + \omega_3 p_{ik} + \omega_4 h_{ik} \tag{7}$$

Where v_{ik}, f_{ik}, p_{ik} and h_{ik} represent browsing, saving, submitting, and interviewing behaviors, respectively, and $\omega_1, \omega_2, \omega_3, \omega_4$ are the behavior weights. To reduce cross-source dimensional differences, this paper normalizes continuous features, embeds categorical features, and applies segmentation, denoising, and vectorization to job-posting text.

2.1.3. *Privacy Threat Model*

This study considers four threats: member inference (determining training participation from model output); attribute inference (inferring salary, location, or background from results); gradient inversion (inferring features from uploaded

gradients); and result inference (inferring profiles or strategies via repeated queries).

This threat model assumes the platform authenticates identities, protects raw logs, and enforces privacy budget policies; enterprise nodes may provide noisy information and attackers may query the interface, but the framework does not verify every company record's truthfulness without external audit.

The adjacent-dataset definition applies at the user level — adding or removing one job seeker and their interaction history should not substantially change the observable recommendation distribution — making the privacy claim user-centered rather than record-centered.

Therefore, this paper introduces a differential privacy mechanism, ensuring that when any adjacent datasets D and D' differ by only one user sample, the recommendation algorithm $\mathcal{M}$ satisfies:

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon} \Pr\left[\mathcal{M}\left(D'\right) \in S\right] \tag{8}$$

Here, ϵ represents the privacy budget; a smaller ϵ results in stronger privacy protection, but may decrease model usability. To accommodate the varying sensitivities of multiple fields in job recommendation, this paper further defines a feature-level privacy budget vector:

$$\boldsymbol{\varepsilon}_i = \left[\varepsilon_s, \varepsilon_e, \varepsilon_c, \varepsilon_b, \varepsilon_p\right] \tag{9}$$

Where ε_b and ε_p correspond to behavioral and preference features, respectively, and are typically set to relatively small values; ε_s and ε_c correspond to skill and certificate information, and can be set to moderate values; publicly available job postings are generally not strongly perturbed.

2.1.4. *Optimization Objective*

The recommendation objective of this study is not simply to maximize accuracy but to jointly optimize recommendation performance, privacy protection, credibility, and system efficiency. The comprehensive objective function is defined as follows:

$$\max F = \alpha Acc + \beta \text{ Trust } - \gamma \text{ PrivacyLoss } - \delta \text{ Latency} \tag{10}$$

Where Acc represents recommendation accuracy, Trust represents recommendation credibility, PrivacyLoss represents privacy leakage risk, Latency represents system response delay, and $\alpha, \beta, \gamma, \delta$ are weight coefficients. In the job recommendation scenario, the system needs to simultaneously satisfy:

$$Acc \geq Acc_{\min}, \quad \text{PrivacyLoss } \leq \rho, \quad \text{Latency } \tau \tag{11}$$

Where, $Acc_{\min}$ is the lower limit of recommendation accuracy, ρ is the privacy leakage risk threshold, and τ is the upper limit of system delay. This multi-objective constraint conforms to the development trend of trusted recommendation systems shifting from accuracy to a comprehensive evaluation of security, reliability, and system availability [15].

2.2. Trusted Computing Recommendation Framework Integrating Differential Privacy

2.2.1. *Overall Architecture of DP-TCERS*

DP-TCERS comprises six layers: data acquisition (collecting profiles, descriptions, records, and company information); privacy protection (differential privacy plus gradient noise); feature representation (mapping structured/text features to a unified vector space); deep matching (computing user-job fit); trusted computing (generating trust scores from source, consistency, and risk); and scalable service (distributed retrieval, caching, and parallel inference).

As Fig. 1 shows, DP-TCERS integrates privacy, trustworthiness, and deployability within one computational framework.

2.2.2. *Differential Privacy Protection Mechanism*

To protect the sensitive features of job seekers, this paper introduces a Laplace mechanism in the feature encoding layer. The original user profile vector is x_i, and the perturbed user profile is:

$$\tilde{x}_i = x_i + \eta \tag{12}$$

In:

$$\eta \sim \text{ Laplace } \left(0, \frac{\Delta f}{\varepsilon}\right) \tag{13}$$

Δf represents the query function sensitivity, and ε represents the privacy budget. For highly sensitive fields such as salary expectations, work location preferences, and application behavior, the system sets a smaller local privacy budget; for moderately sensitive fields such as professional categories and skill tags, the system sets a relatively larger privacy budget to reduce the loss of recommendation accuracy caused by excessive perturbation.

In the model training layer, this paper further introduces gradient clipping and Gaussian noise mechanisms. Let the gradient in the t training round be g_t, and the clipped gradient be:

$$\bar{g}_t = \frac{g_t}{\max\left(1, \frac{\|g_t\|_2}{C}\right)} \tag{14}$$

Where C is the gradient clipping threshold. The gradient after adding Gaussian noise is:

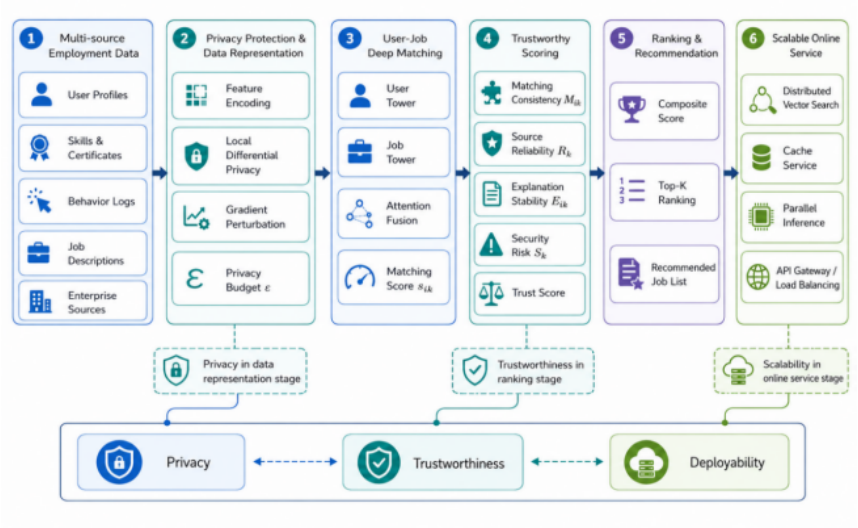

Fig. 1. Overall Architecture of the DP-TCERS Trusted Job Recommendation System.

$$\tilde{g}_t = \bar{g}_t + \mathcal{N}\left(0, \sigma^2 C^2 I\right) \tag{15}$$

Where σ is the noise coefficient. This mechanism reduces the risk of gradient inversion attacks during training. Considering feature- and gradient-layer privacy losses together, this paper defines the overall privacy consumption as:

$$\varepsilon_{\text{total}} = \varepsilon_{\text{feature}} + \sum_{t=1}^{T} \varepsilon_{\text{grad}}^{(t)} \tag{16}$$

To avoid rapidly depleting the privacy budget, the system tracks accumulated privacy losses during training and stops training or raises the noise level when a threshold is exceeded.

The privacy accountant records feature- and training-layer perturbation separately and reports accumulated consumption; the default budget is an experimental configuration under the stated simulator and schedule, not an unrestricted guarantee for arbitrary retraining or querying.

2.2.3. *User-Job Deep Matching Model*

This paper uses a dual-tower structure to model job seeker and job representations; the user tower input is the user profile $\tilde{x}_i$ after privacy perturbation, and the job tower input is the job profile y_k. The user representation is:

$$h_i^u = MLP_u\left(\tilde{x}_i\right) \tag{17}$$

Job representation is defined as follows:

$$h_k^j = MLP_j\left(\text{Encoder}\left(d_k\right) \oplus q_k \oplus l_k \oplus w_k\right) \tag{18}$$

Here, $\text{Encoder}\left(d_k\right)$ is the job-text encoder and $\oplus$ is concatenation; this paper further introduces an attention-fusion mechanism to capture key skills and job requirements. Let the user skill feature set be $\{z_{i1}, z_{i2}, \ldots, z_{ir}\}$, the job requirement feature set be $\{r_{k1}, r_{k2}, \ldots, r_{kl}\}$, and the attention weights be:

$$\alpha_{ab} = \frac{\exp\left(z_{ia}^T w_a r_{kb}\right)}{\sum_{b=1}^{l} \exp\left(z_{ia}^T w_a r_{kb}\right)} \tag{19}$$

User job matching is represented as follows:

$$m_{ik} = \sum_{a=1}^{r} \sum_{b=1}^{l} \alpha_{ab}\left(z_{ia} \odot r_{kb}\right) \tag{20}$$

The final match score is

$$s_{ik} = \sigma\left(W_m\left[h_i^u \oplus h_k^j \oplus m_{ik}\right] + b_m\right) \tag{21}$$

Where σ is the Sigmoid function. The system sorts candidate positions according to s_{ik} and generates Top-K recommendation results. This structure recovers effective matching signals via job-semantic and skill-attention mechanisms even under privacy perturbation.

For cold-start users or new positions, the matching module relies on profile fields, skill tags, job text semantics, and source reliability before enough interaction records accumulate, treating historical interactions as useful signals rather than the sole ranking basis.

2.2.4. *Recommendation Credibility Evaluation Mechanism*

To avoid the system pursuing matching accuracy at the expense of position credibility, this study designs a position recommendation credibility score:

$$\text{Trust}_{ik} = \lambda_1 M_{ik} + \lambda_2 R_k + \lambda_3 E_{ik} + \lambda_4 S_k \tag{22}$$

Where M_{ik} represents the consistency of user-job matching, R_k represents the reliability of job source, E_{ik} represents the explanatory stability, S_k represents the job security risk score, and $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are weighting coefficients. Job-source reliability is computed as:

$$R_k = \theta_1 \text{ Auth }_k + \theta_2 \text{ Fresh }_k + \theta_3 \text{ Comp }_k + \theta_4 \text{ Feed }_k - \theta_5 \text{ Risk }_k \tag{23}$$

Where $Auth_k$ represents the level of enterprise authentication, $Fresh_k$ represents the timeliness of the job posting, Comp_k represents the completeness of the job information, $Feed_k$ represents the quality of historical feedback, and $Risk_k$ represents the risk of anomalies.

Explanatory stability measures whether explanations stay consistent with actual matching factors across similar users and jobs, defined as:

$$E_{ik} = 1 - \frac{1}{L} \sum_{l=1}^{L} \left| e_{ik}^{l} - \hat{e}_{ik}^{l} \right| \tag{24}$$

Where e_{ik}^{l} represents the original explanatory weights, and $\hat{e}_{ik}^{l}$ represents the explanatory weights after perturbation. Higher stability indicates the model maintains consistent recommendation criteria after privacy perturbation.

The credibility score is a ranking signal, not a legal certification: it lowers the ranking weight of incomplete, stale, complained-about, or unauthenticated postings, while confirming enterprise legitimacy still requires platform review, verified credentials, or external audit.

2.2.5. *Scalable Online Recommendation Strategy*

To adapt to large-scale scenarios, DP-TCERS combines offline training (profile updates, job vector indexing, model training, trust-score pre-calculation) and online inference (candidate retrieval, deep re-ranking, trust filtering). Vector retrieval is used in the candidate retrieval phase:

$$C_i = ANN\left(h_i^u, J, K'\right) \tag{25}$$

Where K' is the initial recall size, and $K' > K$. The re-ranking phase is based on the overall score:

$$\text{Score }_{ik} = \mu_1 s_{ik} + \mu_2 \, Trust_{ik} - \mu_3 \, Risk_k \tag{26}$$

The system reduces online overhead by caching popular job vectors, similar-user results, and credibility scores, and uses multi-node parallel inference and load balancing for high-concurrency throughput and latency (Fig. 2).

2.3. Experimental Design

2.3.1. *Dataset Construction and Preprocessing*

This study constructs a simulated dataset covering 10,000 job seekers, 5,000 postings, 80,000 applications, 120,000 browsing entries, and 30,000 bookmarks, with user and job features as detailed in Table 1.

The simulator generates profiles and implicit feedback in three steps: sampling majors, education, skills, and location; assigning postings tags, requirements, and reliability states; and generating interaction events from semantic similarity and popularity, with negatives from non-interactive positions for realistic sparsity. No real personal records were used.

Table 1. Dataset Statistics.

Item	Value
Number of job seekers	10000
Number of positions	5000
Submission History	80000
Browsing History	120000
Favorites History	30000
Skill Tags	50
Industry Category	15
Professional Category	20
Company Source	600

2.3.2. *Feature Description and Privacy Level*

Employment data varies in sensitivity, with perturbation strength set accordingly, as detailed in Table 2.

2.3.3. *Experimental Parameter Settings*

This paper implements the model in Python 3.9 and PyTorch 1.13, with parameters detailed inTable 3.

These are fixed-configuration simulation outputs comparing architectural choices under the same settings, not hardware-level benchmarks or production SLAs.

2.3.4. *Comparison Methods and Evaluation Metrics*

This study compares several baselines against DP-TCERS, detailed in Table 4. All baselines use the same simulated features, candidate pool, split, and Top-K protocol: CF/NCF rely on interaction signals, DeepFM adds feature-crossing, BERT-Rec emphasizes text semantics, and Non-DP Matching uses the same structure without perturbation, with metrics computed on held-out positives after negative sampling.

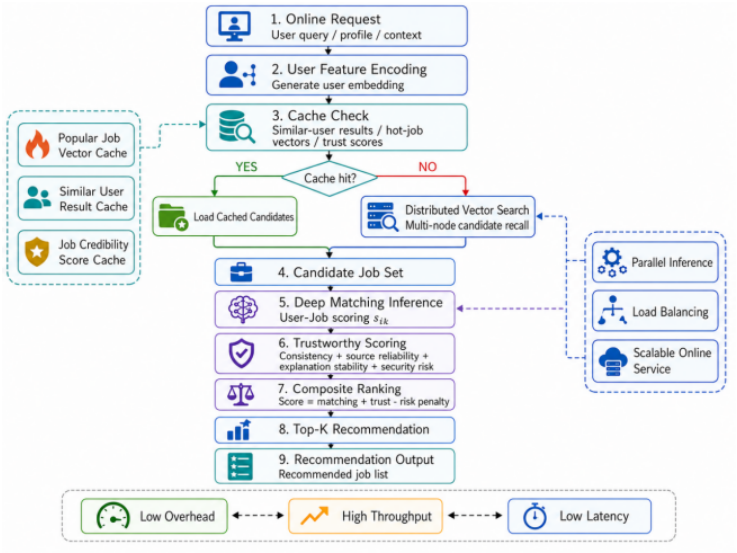

Fig. 2. DP-TCERS Online Recommendation Algorithm Flow.

Table 2. Feature Description and Privacy Level

Feature Type	Feature Description	Privacy Level
Basic Profile	Education level, major, and year of graduation	Medium
Skill Characteristics	Skill tags, certificates, and ability ratings	Medium
Behavioral Characteristics	Browse, save, submit, interview	High
Preference Characteristics	Salary expectations, location, industry preference	High
Job Description	Job title, job description, and job requirements	Low
Company Characteristics	Company type, certification status, source reliability	Medium

Table 3. Experimental Parameter Settings.

Parameter	Value
Embedding Dimension	128
Batch Size	256
Learning Rate	0.001
Optimizer	Adam
Privacy Budget ε	1.0
Gradient clipping threshold C	1.0
Noise coefficient	0.8
Recommendation List Length	10
Number of Training Rounds	50
Distributed Nodes	1-16

3. results and discussion

3.1. Experimental Results and Analysis

3.1.1. Recommendation Performance Comparison

Fig. 3 compares Precision@10, Recall@10, F1@10, and NDCG@10 across methods: CF underuses job descriptions and shows the lowest performance; NCF's non-linear modeling improves on CF; DeepFM's feature fusion further improves Precision@10/NDCG@10; BERT-Rec's semantic representations outperform DeepFM; and Non-DP Matching achieves the highest accuracy. DP-TCERS reaches 0.773, 0.732, 0.752, and 0.765 — slightly below Non-DP Matching but significantly better than CF, NCF, and DeepFM.

This shows DP-TCERS retains high performance despite differential-privacy perturbation, as attention matching and job semantic encoding partly offset the resulting information loss. Since this is a controlled simulation, small gaps between strong baselines should be interpreted cautiously; the main evidence is the joint trend across quality, privacy-risk reduction, trust, and latency.

3.1.2. Top-K Recommendation Stability

Fig. 4 shows Precision@K, Recall@K, and NDCG@K trends for DP-TCERS across list lengths. As K increases from 5 to 30, Precision@K falls from 0.792 to 0.704 while Recall@K rises from 0.681 to 0.812, and NDCG@K declines

Table 4. Comparison Methods and Evaluation Metrics

Category	Content
Methodology	
Baseline Methods	CF, NCF, DeepFM, BERT-Rec, Non-DP Matching
Proposed Method	DP-TCERS
Evaluation Metrics	
Recommendation metrics	Precision@K, Recall@K, F1@K, NDCG@K
Privacy metrics	Privacy leakage risk, attack success rate
Trust metrics	Trust score, recommendation acceptance rate
System metrics	Latency, throughput, scalability efficiency
Visualization	Attention heatmap, t-SNE representation distribution

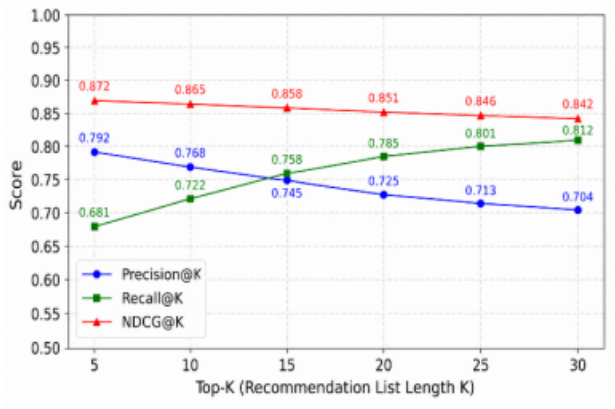

Fig. 3. Comparison of Algorithm Performance of Different Recommendation Methods.

only slightly, keeping well-matched jobs near the top of the list.

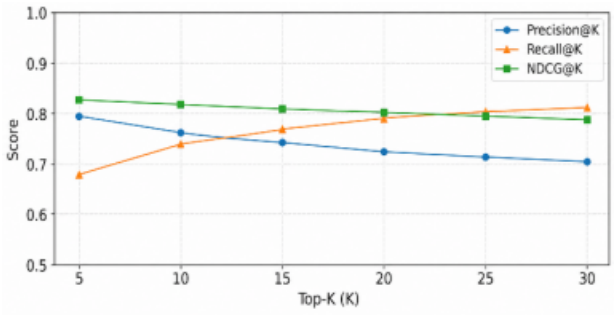

Fig. 4. Recommendation stability of DP-TCERS under different Top-K conditions.

This confirms DP-TCERS remains effective across list lengths, suiting both short curated lists and longer candidate lists.

3.1.3. *Trade-off between Privacy Budget and Recommendation Performance*

Fig. 5 analyzes privacy budget ε' s impact: at $\varepsilon = 0.1$, protection is strongest but noise is high (precision@10=0.684, NDCG@10=0.671); at $\varepsilon = 1.0$, these rise to 0.773/0.765; at $\varepsilon = 8.0$, performance approaches non-DP matching but protection weakens.

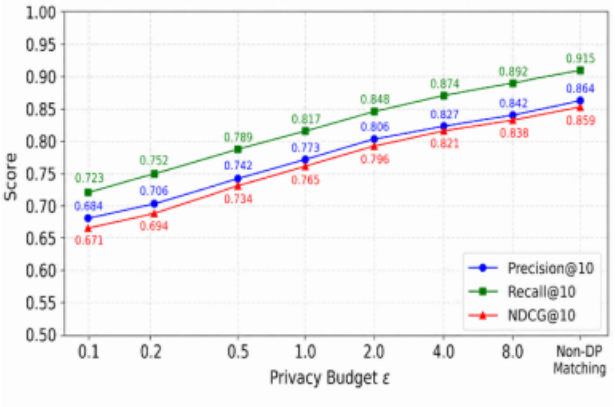

Fig. 5. Impact of Privacy Budget on Recommendation Performance.

This confirms the typical privacy-utility trade-off, leading this study to select $\epsilon = 1.0$ as a balanced default.

3.1.4. *Privacy Leakage Risk Analysis*

Fig. 6 shows privacy leakage risk rising from 0.083 to 0.421 as ϵ increases from 0.1 to 8.0, since a larger privacy budget means weaker perturbation and easier inference of sensitive information.

Combined with Fig. 5, this confirms a clear performance-

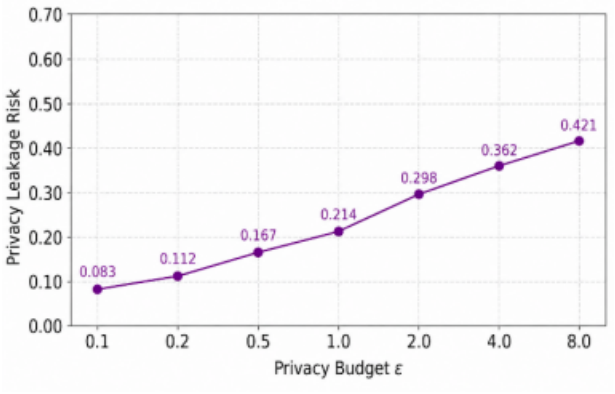

Fig. 6. Privacy Leakage Risk under Different Privacy Budgets.

privacy trade-off, making $\varepsilon = 1.0$ a reasonable balance for employment recommendation.

3.1.5. *Anti-Privacy Attack Capability*

Fig. 7 compares attack success rates of Non-DP, Feature-DP, Gradient-DP, and DP-TCERS. Non-DP shows high leakage (0.438, 0.462, 0.417, 0.391 for the four attack types); Feature-DP reduces profile/behavioral leakage but has limited gradient-inversion defense; Gradient-DP better suppresses gradient inversion but is weaker against attribute inference.

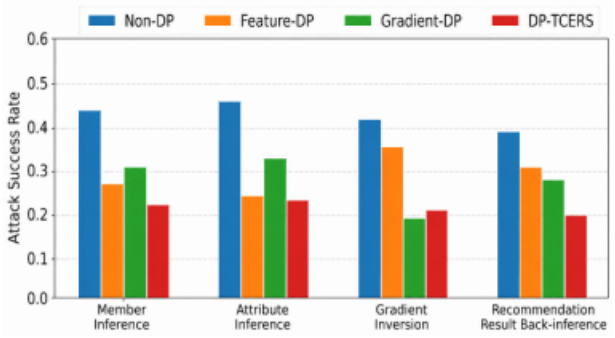

Fig. 7. Comparison of Attack Success Rates under Different Privacy Protection Methods.

DP-TCERS's feature- and training-layer differential privacy achieves the lowest attack success rates across all four attack types (0.219/0.231/0.207/0.198), confirming the dual mechanism improves job recommendation security.

In this experiment, the attack success rate is the fraction of attempts that correctly infer target membership, a sensitive attribute, a reconstructed representation, or a preference category under the simulated attacker — an empirical leakage indicator, not proof that all operational attacks will fail.

3.1.6. *Evaluation of Recommended Job Credibility*

Fig. 8 compares Trust Score distributions: traditional methods average 0.683, under-considering source reliability; accuracy-oriented methods reach 0.721 but lack systematic trust evaluation; DP-TCERS reaches 0.842 with a more concentrated, stable distribution.

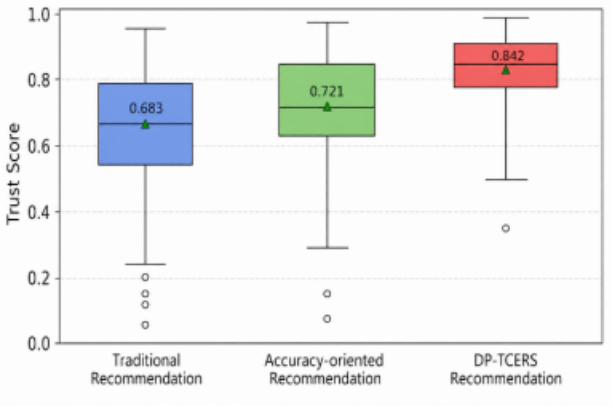

Fig. 8. Comparison of Recommended Job Credibility Distribution.

This shows the trust module effectively filters low-trust jobs, since recommending fake or low-quality postings — even with high matching scores — risks damaging user experience and platform reputation.

3.1.7. *Trust Assessment under Different Job Source Risks*

Fig. 9 shows Trust Score and Recommendation Acceptance Rate across job-source risk levels: low-risk jobs score 0.891 trust and 0.812 acceptance, while unknown-source jobs drop to 0.571 and 0.496, indicating job-source risk significantly affects recommendation acceptance.

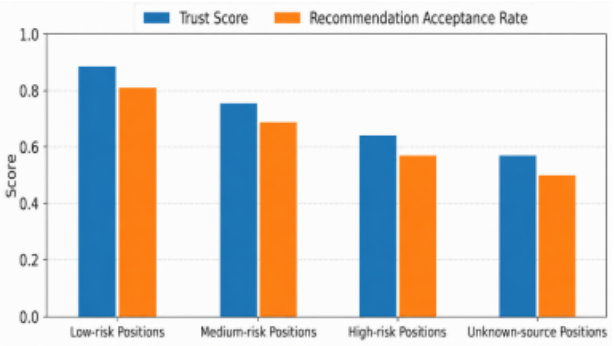

Fig. 9. Trustworthiness Assessment Results under Different Job Source Risks.

DP-TCERS incorporates job-source reliability and security-risk penalties into ranking, avoiding recommending low-trust jobs merely because job text matches user

skills.

3.1.8. System Latency in High-Concurrency Scenarios

Fig. 10 compares average response latency of single-node, distributed, and DP-TCERS-with-cache recommendation across concurrent-user scales. Differences are small at 100 users, but at 10,000 users latency reaches 1486 ms for single-node, 517 ms for distributed, and only 294 ms for DP-TCERS with cache.

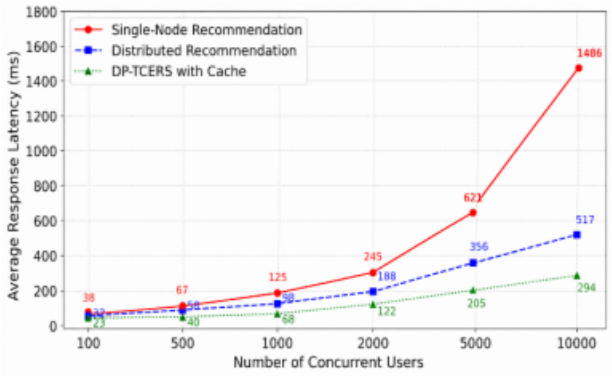

Fig. 10. Average system response latency for different numbers of concurrent users.

This shows single-node deployment cannot scale to high concurrency, while DP-TCERS cuts latency via distributed retrieval, caching, and parallel inference — crucial during graduation-season surges.

The latency comparison used the same request pattern and retrieval configuration for all systems; values are service-simulation estimates requiring validation on a concrete deployment stack before production use.

3.1.9. Ablation Experiments

Fig. 11 shows performance after removing each module: removing DP slightly raises NDCG@10 (0.765→0.773) but sharply raises attack success (0.219→0.438); removing trust evaluation drops Trust Score (0.842→0.706); removing attention matching drops NDCG@10 to 0.733 ; removing caching raises latency (74 → 126 ms); and removing source reliability drops Trust Score to 0.751.

Each module thus plays a distinct role: differential privacy reduces attack risk, trust assessment improves reliability, attention matching improves accuracy, and caching/distributed service improves response efficiency.

3.1.10. Noise Scale Sensitivity Analysis

Fig. 12 shows the noise scale's impact: as it increases from 0.2 to 1.2, Precision@10 falls from 0.784 to 0.724 while attack success rate falls from 0.362 to 0.181 — more noise strengthens privacy but weakens matching.

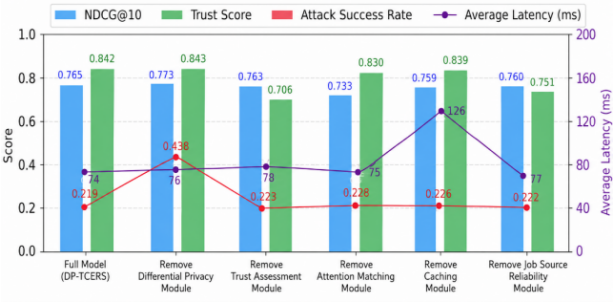

Fig. 11. Overall performance comparison under different module ablation conditions.

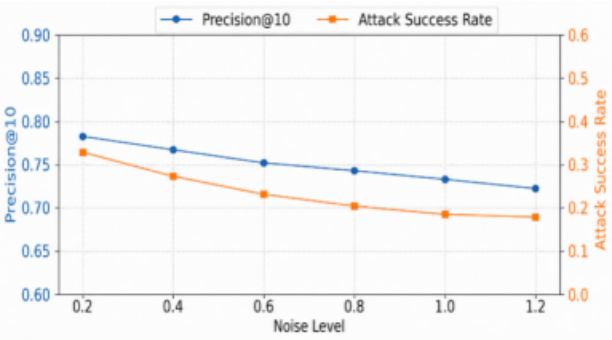

Fig. 12. Impact of Noise Scale on Recommendation Accuracy and Attack Success Rate.

Between 0.6 and 0.8, a good balance is achieved between performance and defense, so we selected a noise coefficient of 0.8 — confirming DP-TCERS's privacy parameters are chosen through sensitivity experiments rather than arbitrarily.

3.2. Discussion

3.2.1. Balance between Privacy Protection and Recommendation Utility

Differential privacy causes some acceptable performance loss: DP-TCERS's Precision@10 drops from 0.786 to 0.773 and NDCG@10 from 0.773 to 0.765 versus Non-DP Matching, while member-inference success drops from 0.438 to 0.219 — a worthwhile trade given how sensitive job seekers' preferences and salary expectations are.

3.2.2. The Importance of Trustworthy Recommendations for Employment Platforms

Unlike entertainment or product recommendations, job recommendation directly affects careers and recruitment efficiency, so source reliability and security-risk identification matter. DP-TCERS's Trust Score significantly exceeds traditional and accuracy-oriented baselines, suggesting popular positions should not dominate at the expense of unpopular but well-matched ones.

Employment recommendations also require group-level fairness monitoring — examining whether different majors, education levels, regions, or job-intention groups receive comparable exposure to qualified positions, and combining trust scoring with fairness constraints to avoid disadvantaging less-represented groups.

3.2.3. Scalable Information System Attributes

This study models job recommendation as a scalable information system: DP-TCERS maintains low latency at 10,000 concurrent users with throughput scaling by node count, confirming deployment feasibility beyond offline validation.

3.2.4. Applicable Boundaries and Improvement Directions

This paper has limitations: data are simulator-generated and need real-platform validation; values lack repeated-run confidence intervals; the framework covers differential privacy and trusted scoring but not federated learning or trusted execution; fairness constraints are conceptual; source-reliability scoring cannot verify factual accuracy; and future systems should detect generative-AI content while retaining privacy and auditing [15].

4. Conclusions

This study proposes DP-TCERS, a differential-privacy trusted computing framework for industrial talent recommendation. Key conclusions:

- DP-TCERS is designed for large-scale employment systems, jointly addressing privacy, matching accuracy, credibility, and scalability.
- Feature- and training-layer differential privacy reduces leakage risk from profiles, behavior, and gradients, while attention-based dual-tower matching improves effectiveness.
- A trust score combining source reliability, matching consistency, interpretability stability, and security risk strengthens credibility, aided by distributed retrieval, caching, and parallel inference.
- DP-TCERS maintains high accuracy while significantly reducing privacy-attack success and latency under high concurrency, providing secure, reliable, scalable support for industrial talent recruitment and public/corporate platforms.
- Future research will combine real employment data, federated learning, and trusted auditing to improve practicality in cross-institutional scenarios. These results are framework-level validation; external data validation, repeated benchmarks, and governance procedures are still required before use in high-stakes employment decisions.

References

[1] Y. Mashayekhi, N. Li, B. Kang, J. Lijffijt, and T. De Bie, (2024) *"A Challenge-based Survey of E-recruitment Recommendation Systems"* **ACM Computing Surveys** ***56***(10): 1–33. DOI: 10.1145/3659942.

[2] F. P. Poncio, (2024) *"Navigating Techniques in Job Recommender Systems on Internship Profile Matching: A Systematic Review"* **Journal of Research in Innovative Teaching & Learning** ***17***(2): 352–367. DOI: 10.1108/JRIT-01-2024-0016.

[3] S. A. Alsaif, M. S. Hidri, H. A. Eleraky, I. Ferjani, and R. Amami, (2022) *"Learning-based Matched Representation System for Job Recommendation"* **Computers** ***11***(11): 161. DOI: 10.3390/computers11110161.

[4] R. H. El-Deeb, W. Abdelmoez, and N. El-Bendary, (2025) *"Enhancing E-Recruitment Recommendations Through Text Summarization Techniques"* **Information** ***16***(4): 333. DOI: 10.3390/info16040333.

[5] D. Roy and M. Dutta, (2022) *"A Systematic Review and Research Perspective on Recommender Systems"* **Journal of Big Data** ***9***: 59. DOI: 10.1186/s40537-022-00592-5.

[6] Y. H. Alfaifi, (2024) *"Recommender Systems Applications: Data Sources, Features, and Challenges"* **Information** ***15***(10): 660. DOI: 10.3390/info15100660.

[7] S. Wang, X. Zhang, Y. Wang, and F. Ricci, (2024) *"Trustworthy Recommender Systems"* **ACM Transactions on Intelligent Systems and Technology** ***15***(4): 84:1–84:20. DOI: 10.1145/3627826.

[8] Y. Wang, W. Ma, M. Zhang, Y. Liu, and S. Ma, (2023) *"A Survey on the Fairness of Recommender Systems"* **ACM Transactions on Information Systems** ***41***(3): 52:1–52:43. DOI: 10.1145/3547333.

[9] Z. Xu, H. Zeng, J. Tan, Z. Fu, Y. Zhang, and Q. Ai, (2023) *"A Reusable Model-Agnostic Framework for Faithfully Explainable Recommendation and System Scrutability"* **ACM Transactions on Information Systems** ***42***(1): 29:1–29:29. DOI: 10.1145/3605357.

[10] P. Vahdatian, M. Latifi, and M. Ahsan, (2025) *"Designing Trustworthy Recommender Systems: A Glass-Box, Interpretable, and Auditable Approach"* **Electronics** ***14***(24): 4890. DOI: 10.3390/electronics14244890.

[11] Z. Xu, C. Chu, and S. Song, (2024) *"An Effective Federated Recommendation Framework with Differential Privacy"* **Electronics *13***(8): 1589. DOI: 10.3390/electronics13081589.

[12] H. Zhang, F. Luo, J. Wu, X. He, and Y. Li, (2023) *"LightFR: Lightweight Federated Recommendation with Privacy-Preserving Matrix Factorization"* **ACM Transactions on Information Systems *41***(4): 90:1–90:28. DOI: 10.1145/3578361.

[13] J. Neera, X. Chen, N. Aslam, K. Wang, and Z. Shu, (2023) *"Private and Utility Enhanced Recommendations With Local Differential Privacy and Gaussian Mixture Model"* **IEEE Transactions on Knowledge and Data Engineering *35***(4): 4151–4163. DOI: 10.1109/TKDE.2021.3126577.

[14] J. Luo, X. Yang, X. Yi, and F. Han, (2023) *"Privacy-preserving Recommendation System Based on User Classification"* **Journal of Information Security and Applications *79***: 103630. DOI: 10.1016/j.jisa.2023.103630.

[15] Y. Ge, S. Liu, Z. Fu, J. Tan, Z. Li, S. Xu, Y. Li, Y. Xian, and Y. Zhang, (2025) *"A Survey on Trustworthy Recommender Systems"* **ACM Transactions on Recommender Systems *3***(2): 13:1–13:68. DOI: 10.1145/3652891.