Overview of TalentCLEF 2026: Task A — Contextualized Job–Person Matching

Hermenegildo Fabregat[1,2,*,†], Laura García-Sardiña[1,†], Paula Estrella[1,†], Luis Gasco[1,†], Casimiro Pio Carrino[1], Daniel Deniz[1], Alvaro Rodrigo[2] and Rabih Zbib[1]

[1]*Avature Machine Learning, Spain*
[2]*NLP & IR Group at UNED, Madrid, Spain*

Abstract

This paper presents an overview of Task A of TalentCLEF 2026, reformulated this edition as a contextualized person–job matching problem: given a job vacancy, systems must rank a pool of candidate résumés by their suitability for the vacancy. We describe the task definition, the synthetic multilingual (English and Spanish) dataset, and the three evaluation settings—multilingual, cross-lingual, and bias-controlled. We then analyze the 11 participating systems that contributed working notes, characterizing them along five recurring axes: multi-stage retrieval, reranking and rank fusion, the use of Large Language Models, taxonomy- and graph-based knowledge, and training strategies. We report the official results for all three settings and complement them with a post-hoc meta-analysis of the headroom available through per-query system fusion.

Keywords

Natural Language Processing, Human Capital Management, Human Resources, Multilinguality, Cross-linguality, Skill Predictions, Job Title Ranking

1. Introduction

The labor market is shifting towards skills, rather than job titles or credentials, as its primary value unit, with a large share of the workforce expected to reskill or change roles in the coming years [1, 2, 3]. This places Human Capital Management (HCM) under growing pressure, as a single vacancy can attract hundreds of applications [4] that are infeasible to review manually at scale. Natural Language Processing and Information Retrieval are therefore central to modern HCM, yet the domain remains underserved by shared, reproducible benchmarks that also reflect its inherently multilingual nature and its direct impact on people's employment opportunities. TalentCLEF was established to address this gap, providing an open evaluation framework for skill and job-title intelligence in HCM [5, 6]. The 2026 edition comprises two complementary tasks: contextualized job–person matching, addressed in this overview, and job–skill matching with skill type classification, described in the corresponding Task B overview [7].

For the 2026 edition,[1] Task A has been reformulated from the previous *job-title matching* benchmark into a problem of *contextualized job–person matching*: given a job vacancy, systems must rank a pool of candidate résumés by their suitability for the vacancy. This moves the task from short, isolated strings toward full documents and is evaluated under three complementary settings.

This overview makes three contributions. First, we describe TalentCLEF 2026 Task A and the three evaluation settings (Section 2). Second, we describe the dataset generation pipeline and its final statistics (Section 3). Third, we analyze the 11 participating systems that contributed working notes, characterizing them along five recurring axes: multi-stage retrieval, reranking and rank fusion, the use

CLEF 2026 Working Notes, 21–24 September 2026, Jena, Germany
*Corresponding author.
†These authors contributed equally.
machinelearning@avature.net (H. Fabregat)
0000-0001-9820-2150 (H. Fabregat); 0000-0003-4592-8884 (L. García-Sardiña); 0000-0002-4976-9879 (L. Gasco); 0000-0002-1442-3672 (C. P. Carrino); 0000-0002-0313-2127 (D. Deniz); 0000-0002-6331-4117 (A. Rodrigo); 0000-0002-7140-3048 (R. Zbib)

[1]TalentCLEF 2026 is organized as a lab at CLEF 2026.

of Large Language Models, taxonomy- and graph-based knowledge, and training strategies (Section 4). Then, we report the official results and complement them with a post-hoc meta-analysis of the headroom available through per-query system fusion (Section 5). Finally, in Section 6, conclusions are drawn, and some ideas for future editions are outlined.

2. Task A: Contextualized Job–Person Matching

2.1. Motivation and Background

Candidate matching is one of the central challenges in Human Capital Management (HCM). When performed manually, the process relies on recruiters reading individual résumés, interpreting job descriptions, and applying their domain expertise and judgment to decide which candidates best fit a vacancy. Although this manual approach naturally incorporates human reasoning and contextual knowledge into the decision, it scales poorly in the contemporary labor market, where a single vacancy may attract hundreds of applications [4]. As application volumes grow, an exhaustive manual review becomes infeasible, introducing both bottlenecks and inconsistency into the hiring pipeline.

To address these limitations, automatic matching systems grounded in Natural Language Processing (NLP) and information extraction have been developed over the past decade. These systems typically identify and normalize the relevant entities found in vacancies and candidate profiles, such as job titles, skills, competencies, education, and languages, among others, and then compare the entities shared between the two types of documents to estimate suitability [8, 9, 10]. This entity-centric paradigm is well established and effective, but tends to reduce a candidate to a bag of normalized attributes, discarding much of the contextual signal contained in the original text.

The recent emergence of Large Language Models (LLMs) presents the matching problem from a richer, more contextual perspective. LLM-based approaches can reason over additional dimensions without task-specific fine-tuning, including seniority level, evidence of expertise behind a claimed skill, demonstrated experience in real-world settings, and the inference of implicit or related competencies [11, 12, 13]. These capabilities open the door to matching systems that are more flexible and context-aware than the entity-matching pipelines that preceded them, while simultaneously raising new questions about fairness, robustness, and cross-lingual transfer that a shared-task setting is well positioned to examine.

2.2. From Job-Title Matching to Contextualized Matching

In the previous edition of TalentCLEF [5, 6], Task A was framed exclusively as a *job-title matching* problem, in which the goal was to relate occupational titles to each other. Although useful as a controlled benchmark, job-title matching captures only a narrow slice of the information that recruiters actually weigh: candidate matching in practice draws on far richer descriptions of both the vacancy and the candidate, including responsibilities, accumulated experience, skill evidence, and educational background.

For the TalentCLEF 2026 edition [14, 15], Task A has therefore been reformulated as a broader problem of *contextualized job–person matching*. Given a job vacancy, the objective is to identify and rank the most suitable candidates drawn from a pool of résumés. This reformulation moves the task from short, isolated strings to full documents, shifting the modeling challenge from lexical or embedding-level title similarity toward a holistic assessment of how well a candidate profile satisfies the explicit and implicit requirements of a vacancy. TalentCLEF 2026 comprises two tasks; in this overview, we focus exclusively on Task A.

Participants are free to approach the task with any methodological strategy, including information extraction, prompt engineering with LLMs, dense or sparse information retrieval, learning-to-rank or hybrid combinations thereof. The expected output in all cases is a ranked list of candidates for each job vacancy, which aligns the task with the standard formulation of an ad-hoc retrieval problem and enables evaluation with established ranking metrics.

2.3. Task Definition

We formalize Task A as a ranking problem. Let V denote a job vacancy (the *query*) and let $\mathscr{C} = \{c_1, c_2, \ldots, c_n\}$ denote the pool of candidate résumés (the *corpus*). For each vacancy V, participating systems must produce a ranking π_V over $\mathscr{C}$ that orders candidates by their estimated suitability for V, such that résumés judged relevant by expert annotators are ranked above non-relevant ones. The task is intentionally agnostic to the internal mechanism that produces π_V: systems may operate at the level of extracted entities, full-text representations, or LLM-generated relevance judgments, provided they emit a complete ranked list.

This formulation supports three complementary evaluation scenarios, described in Section 2.4: a *multilingual* setting in which vacancies and résumés share a language, a *cross-lingual* setting in which they differ, and a *bias* setting in which ranking stability between demographic groups is assessed.

2.4. Evaluation

The task was evaluated through a competitive event hosted on Codabench[2], providing the participating teams with a shared environment to submit predictions and consult the official scoreboard. Hosting the task on Codabench also keeps it available as an open benchmark for continuous evaluation after the official challenge concludes, supporting reproducibility and the comparison of future systems against the original participant pool.

This edition considers three evaluation settings, each targeting a distinct aspect of contextualized matching:

Multilingual. Both the job vacancy and the candidate résumés are written in the same language. This is the main matching scenario and is independently evaluated for English and Spanish.

Cross-lingual. The job vacancy is written in English, while the résumés are written in Spanish. This setting reflects multilingual workplaces, where a company may advertise a vacancy in one language while candidates describe their experience in another, and it stresses a system's ability to align meaning across languages rather than match surface forms.

Bias. Because matching systems directly affect people's employment opportunities, it is essential to examine not only aggregate performance but also behavior across demographic groups such as gender and ethnicity. In this setting, we assess whether systems produce consistent rankings irrespective of the gender of the candidate.

For multilingual and cross-lingual scenarios, system performance is measured with Mean Average Precision (MAP) over the ranked list of candidates, which rewards systems that place relevant candidates near the top of the ranking. For the bias scenario, we adopt Rank-Biased Overlap (RBO) to quantify gender bias: by comparing the rankings produced for otherwise-equivalent candidates differing in gender, RBO captures the extent to which a system's ordering is stable under that demographic perturbation, with higher overlap indicating more equitable behavior. Together, these metrics let us characterize each system along two orthogonal axes, retriever effectiveness and fairness, rather than collapsing them into a single score.

Organizers' baseline. To provide a reference point for all submitted systems, the organizers released as baseline a zero-shot dense-retrieval system based on `sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2`, a 12-layer multilingual sentence encoder with 118M parameters. For each of the three evaluation directions (EN-EN, ES-ES, and EN-ES), job-offer queries and résumé corpus documents are encoded independently; candidates are ranked by cosine similarity of their respective embeddings without any task-specific fine-tuning.

[2]Task A Codabench: https://www.codabench.org/competitions/14226/

3. Dataset Generation Pipeline

For Task A, we provide a multilingual dataset for contextualized job−-person matching that covers both English and Spanish. The corpus is divided into a development set and a test set. Training data were not distributed; instead, participating teams were allowed to use any external resources or additional information they considered relevant, an arrangement that reflects realistic deployment conditions and encourages diverse modeling strategies rather than convergence on a single supervised recipe.

To create realistic job records and résumés for the job–person matching task, we developed a comprehensive synthetic data generation pipeline, illustrated in Figure 1. The main objective was to construct a high-quality, stylistically diverse, and privacy-preserving dataset of résumés and job offers without relying on proprietary human profiles. As source data, we used internal datasets in which candidates and jobs are represented by a job title and an associated set of skills.

The final pipeline consists of five distinct stages: (1) synthetic résumé generation with placeholders derived from titles and skill sets, in English; (2) placeholder filling to inject realistic personal demographic details; (3) synthetic job offer generation to produce generic, platform-agnostic job postings, in English; (4) résumé and job offer translation into Spanish; and (5) data annotation and quality revision.

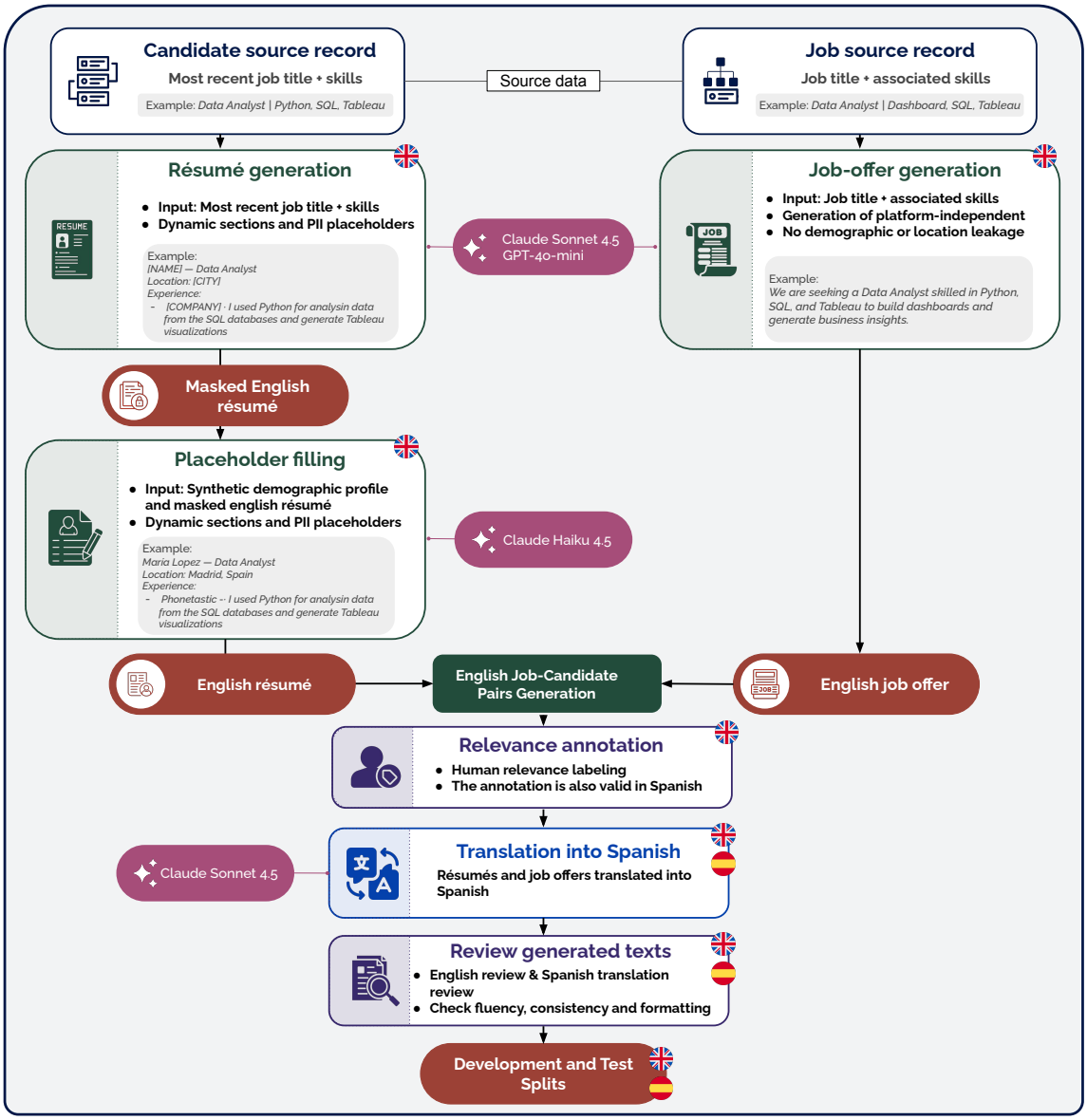

Figure 1: Task A dataset generation pipeline.

3.1. Résumé Generation

The first stage of our pipeline centers on the generation of candidate résumés. To ensure high stylistic diversity and structural variation across the generated outputs, we employed a multifaceted strategy: configuring models with a high temperature, utilizing two distinct large language models (LLMs) comprising `claude-sonnet-4-5` and `gpt-4o-mini`, and applying dynamic prompt instructions.

For the test set, résumés were generated by conditioning the LLMs on a candidate's most recent job title, their reported skills, and associated job contexts. Specifically, the prompt included up to five matching and five non-matching job titles sampled from existing annotations. While matching jobs were selected randomly, non-matching jobs were deliberately sampled based on descending skill overlap. This approach presented the generative model with highly challenging negative examples, forcing it to differentiate subtle domain boundaries. For the development set, the generation step followed the same pattern, except that positive and negative job titles were not included as part of the input. This was due to the lack of annotations available for the development source data. Instead, generation was based solely on job title and skills.

We use a dynamic prompt dictating that the models must analyze the input skills, filter out inconsistencies, and synthesize a cohesive work experience, education history, and relevant optional sections. The structure was dynamically randomized by changing the exact list of sections to include; prompts included a core set of necessary sections (Contact, Work Experience, Education, and Skills) paired with a randomly selected subset of optional sections (such as objectives, projects, certificates, languages, hobbies, publications, achievements, or others). Furthermore, section names were randomized within the instructions (e.g., the work section could be specified as "Work Experience", "Professional Experience", or "Work History"), and the model was explicitly instructed that it could modify the given names to fit the profile style.

To satisfy privacy constraints, placeholders were strictly forced on all personal and sensitive information. The prompt made it explicit that it should not be possible to deduce personal or demographic information from the masked resume text.

3.2. Placeholder Filling and Profile Injection

To transform masked resumes into realistic profiles while protecting privacy, all structural placeholders (e.g., `[NAME]`, `[COMPANY]`, `[CITY, STATE]`) were systematically filled using `claude-haiku-4-5`.

For each masked résumé, three distinct demographic profiles were generated based on gender (*male*, *female*, and a third non-conforming option) and a randomly sampled place of origin. A varied list of countries grouped by region guided the model to generate realistic and geographically consistent names, addresses, phone numbers, and languages. The model was strictly instructed to preserve the core professional content, adjusting only gender-marked professional titles or conflicting language proficiencies, where necessary. For the non-conforming gender profile, we evaluated both *diverse* and *non-binary* configurations. However, quality analysis revealed systemic algorithmic biases: the model consistently generated highly repetitive names (e.g., "Alex" for Western regions) or erroneously assigned clearly gendered names (e.g., "Alejandro"). Due to this lack of robust cross-regional representation for non-binary identities, these filled résumés were excluded from the final dataset.

3.3. Job Offer Generation

Job offers were synthesized using `gpt-5-nano`. The prompt design enforced platform-independence and structural neutrality, requiring the model to rewrite original job titles to strip out localized or superfluous text, mention the title naturally, and focus exclusively on core competencies. Crucially, the models were restricted from introducing demographic markers, location details, salary brackets, or explicit years of experience, ensuring that the data remained generic and free of demographic leakage.

3.4. Translation into Spanish

Following the generation of the English datasets, the Spanish counterparts were produced via a zero-shot, LLM-based approach powered by `claude-sonnet-4-5`. We designed document-specific prompts to enforce localized translation conventions while preserving structural fidelity and professional tone. Technical terms, frameworks (e.g., ".NET", "Azure", "Git"), and objective entities (names, dates, URLs) were preserved verbatim. For resumes, the candidate's gender profile was explicitly provided in the context to ensure grammatically accurate, gender-marked translation. Conversely, the job description prompts strictly mandated the use of gender-neutral and inclusive Spanish phrasing. Finally, structural alignment was verified via a paragraph-count consistency check between the source text and its target translation.

3.5. Annotation and Validation

Finally, human experts conducted a joint annotation and verification phase on the data in English, labeling job-candidate pairs as relevant or non-relevant while simultaneously performing a rigorous quality check of the synthesized text. Any identified errors or linguistic inconsistencies were flagged and manually corrected. For the Spanish translations, annotations from the original job-person pairs were preserved, therefore, this phase included only a light quality review of the translations.

Table 1 summarizes the main statistics of the final dataset, reporting the number of job descriptions (queries) and résumés (corpus) per language and partition. Data were made available to participating teams through Zenodo.[3]

Table 1: Statistics of Task A's development and test sets by language.

Split	Language	Queries	Corpus
Development	English	10	472
	Spanish	10	472
Test	English	40	476
	Spanish	40	476

The contrast between the development and test partitions is deliberate: the development set offers a small number of queries (10 per language) for system tuning and qualitative inspection, while the test set expands the query set fourfold (40 per language) over a corpus of comparable size, providing a more robust basis for the final comparison of systems.

4. Participants and Systems

Task A attracted 91 registered participants in this edition. During the evaluation phase, 21 teams submitted at least one run, producing a total of 384 submissions across the development and test sets. Of these, 11 teams contributed working notes describing their systems, which constitute the basis of the analysis presented below. Table 2 provides short descriptions of the approaches proposed by these teams. For each system, the summary highlights the main representation and retrieval strategy, the use of reranking or rank fusion, relevant training or architectural details, the role of LLMs, and, where applicable, the use of external data, knowledge graphs, or explicit bias-mitigation mechanisms.

[3]Task A corpus: https://doi.org/10.5281/zenodo.17625261

Table 2: Overview of Task A systems described in participating teams' working notes.

Team	Ref	System description
qwerity	[16]	**Two-stage dense retrieval with sparse neural fusion.** JobBERT-v3 retrieves the top 100 candidates, while SPLADE produces a complementary sparse ranking; both rankings are combined using weighted reciprocal rank fusion. A multilingual Jina-ColBERT-v2 late-interaction reranker is evaluated as an alternative, but the SPLADE-based configuration is the primary submitted system.
CYUT	[17]	**Multi-view hybrid retrieval with generative document expansion.** The system combines BGE-M3 dense retrieval, BM25, forward HyDE, reverse HyDE, and a concatenated job–HyDE view. Gemma 3 27B generates hypothetical CVs and job descriptions, with forward HyDE enriched using ESCO and O*NET concepts. The five rankings are fused using weighted RRF; RankZephyr reranks English results, while the Spanish and cross-lingual configurations omit reranking and use translation where required.
classum	[18]	**Language-specific ensemble over semantically enriched job and résumé views.** Job descriptions and résumés are converted into structured representations covering roles, seniority, domain, skills, tasks, and responsibilities. Dense and Gemini-based embedding scores are combined with GPT-5.5 and Claude Opus 4.6 listwise scores and Gemini rubric-based compatibility scores. Equal-weight score fusion is followed by role-group smoothing and local Zerank2 reranking within score plateaus.
DevoMatcher	[19]	**Hybrid lexical–dense retrieval with ESCO calibration and LLM reranking.** BM25, BGE-M3, and SkillMatch MPNet rankings are combined using weighted RRF. ESCOXLM-R extraction, ESCO skill mining, and Node2Vec graph signals provide taxonomy-aware calibration. GPT-5.2 performs adaptive, ESCO-grounded listwise reranking over the top 50 candidates, with demographic cues masked before all scoring stages.
Skillberg.app	[20]	**Knowledge-graph-grounded dense–sparse retrieval with language-conditional reranking.** A multilingual E5 bi-encoder is contrastively fine-tuned through a six-stage curriculum derived from a proprietary cross-framework occupation–skill graph, using GISTEmbedLoss and MatryoshkaLoss. Corpus-side representations are enriched offline with Mistral-generated descriptions, and dense similarity is combined with BM25. A Jina-v2 cross-encoder reranks Spanish results but is omitted for English because it degraded development performance.
HR_NLP	[21]	**Cross-lingual teacher–student bi-encoder.** JobBERT acts as the teacher and multilingual MPNet as the student. The student is trained on job–positive-résumé–negative-résumé triplets using cross-entropy, triplet-margin, and KL-divergence distillation losses, followed by iterative hard-negative mining to distinguish candidates with similar skills but different experience levels. External Kaggle data are reformatted with Claude and translated into Spanish with MarianMT; no LLM is used during ranking.
UBCS	[22]	**Hybrid dense–sparse retrieval with offline document expansion.** A multilingual MiniLM bi-encoder provides dense retrieval, while a T5-based Doc2Query model generates document expansions that are indexed with BM25 in PyTerrier. The union of the dense and sparse candidate sets is ranked through equal-weight late score fusion after per-query min–max normalization. T5 is used only for offline expansion, with no neural or LLM reranking stage.

Continued on the next page

Table 2 – continued from the previous page

Team	Ref	Methods
VerbaNex	[23]	**Hybrid dense–lexical retrieval with cross-encoder reranking.** BGE-M3 dense similarity and BM25 lexical scores are min–max normalized and combined through weighted score fusion. Job descriptions are augmented through structured, rule-based query enrichment that makes skills, experience, education, and other requirements explicit. The top 700 candidates are rescored with a multilingual mMiniLMv2 cross-encoder, and the final top 100 are returned.
ui-nlp	[24]	**Multilingual bi-encoder score ensemble.** For the English and Spanish monolingual settings, mE5-large-instruct and mE5-large independently encode job descriptions and résumés, and their per-query min–max-normalized cosine-similarity scores are fused using language-specific weights selected on the development set. The cross-lingual run uses mE5-large-instruct alone because no corresponding development judgments were available. The submitted system uses neither cross-encoder nor LLM reranking.
bipboopbipboop	[25]	**Multi-stage neural scoring with pointwise and listwise LLM reranking.** For English, scores from BGE-LoRA, MiniLM, Qwen3-Reranker-8B, and a Qwen3-235B rubric-based LLM judge are combined through geometric-mean fusion. The fused top 60 candidates enter a three-round Qwen3-235B listwise tournament, whose final round applies sliding-window reranking to the top 15. Spanish and cross-lingual runs fuse BGE-M3 with the pointwise reranker before applying the same tournament.
QTPride	[26]	**LLM-parsed multi-view dense retrieval with graph-enhanced fusion.** GPT-5 mini converts job descriptions and résumés into structured JSON. Multilingual E5 embeddings provide full-text, work-experience, explicit-skill, and education views, while a self-supervised R-GCN over an ESCO occupation–skill graph produces a structure-aware skill view. The best submitted run combines the full-text, work-experience, and graph rankings using weighted RRF; the LLM is used for document parsing rather than reranking.

Abbreviations: RRF = reciprocal rank fusion; HyDE = hypothetical document embeddings; R-GCN = relational graph convolutional network.

The first observation is that, despite the absence of task-specific training labels, participation was broad and methodologically diverse. Because no supervised relevance judgments were released for training, teams converged on a shared high-level strategy —treating Task A as an ad-hoc retrieval problem solved with reusable, largely pretrained components— while differing substantially in how they enriched, combined and reranked the underlying signals. The following subsections organizes these systems along five recurring axes: the multi-stage retrieval backbone (Section 4.1), the use of reranking and rank fusion (Section 4.2), the role of LLMs (Section 4.3), the incorporation of structured taxonomic and graph knowledge (Section 4.4), and the training and fine-tuning strategies adopted (Section 4.5). Finally, Section 4.6 discusses how bias was addressed.

4.1. Multi-Stage Retrieval Architectures

The solutions proposed this year are predominantly based on multi-stage information retrieval architectures. Participants consistently frame the problem as a retrieval task: they first generate an initial set of candidates using diverse textual representation methods and, in many cases, apply a subsequent reranking stage to refine the final ranking.

In the retrieval stage, models that produce dense semantic representations constitute the core component of nearly every system. Multilingual bi-encoders are the dominant choice, with `multilingual-e5-large-instruct` and `BGE-M3` appearing across the majority of submissions (*ui-nlp*, *VerbaNex*, *CYUT*, *DevoMatcher*, *Skillberg.app*, and *QTPride*, among others) as the backbone for cross-lingual matching between English and Spanish. These dense representations are frequently complemented with lexical

methods such as BM25, which appears in several submissions as an exact-matching mechanism. This combination is particularly useful in the HCM domain, where normalized skill names, certifications, named tools, and labor-market expressions carry the decisive signal that purely semantic encoders tend to smooth over and where retrieving a strong initial candidate set depends on capturing this precise terminology. In addition, models from the JobBERT family [27] are used by teams such as *qwerity*, *classum*, and *HR_NLP*, seeking to exploit encoders adapted to the labor-market domain rather than general-purpose embedding models. *qwerity*, in particular, adopts JobBERT as a first-stage dense retriever and systematically contrasts it with reranking alternatives.

A recurring design decision concerns how the cross-lingual (en–es) condition is handled. Whereas several teams rely on the inherent multilingual alignment of their encoders, others adopt an explicit *translation-to-English anchor* strategy, translating Spanish résumés into English before indexing so that lexical routes such as BM25 can operate in a shared language space; *CYUT* is a representative example that has translated the Spanish corpus prior to retrieval. As discussed in Section 5, the relative success of these two strategies is one of the more informative contrasts of this edition.

4.2. Reranking and Rank Fusion

Another frequent pattern is the incorporation of a reranking stage applied to the retrieved candidates. Seven of the 11 teams explicitly include some form of reranking. *qwerity*, for example, explores complementary strategies by combining the sparse lexical model SPLADE [28] with late-interaction reranking based on the ColBERT architecture [29], reporting that SPLADE-based reciprocal-rank-fusion reranking yields the strongest overall configuration while ColBERT remains competitive in Spanish. Other teams, such as *VerbaNex*, *ui-nlp*, and *Skillberg.app*, use pretrained or fine-tuned cross-encoders to improve the precision of the final ranking. Notably, *Skillberg.app* applies its cross-encoder reranker *conditionally*, enabling it only on the languages where it improved the bi-encoder baseline in internal evaluation and falling back to the first-stage ranking elsewhere —an explicit acknowledgment that reranking is not uniformly beneficial across languages.

Because participants often combine several representations to produce the final ranking, rank-fusion methods are central to many systems. Techniques including reciprocal rank fusion, weighted RRF, late (score) fusion with min–max normalization, and geometric-mean fusion are used to aggregate evidence from different ranked lists. This allows systems to combine complementary strengths: lexical models capture exact terminology, dense encoders capture semantic similarity, cross-encoders refine relevance estimation, and LLM-based rerankers —applied in some systems before and in others after fusion— provide finer-grained relevance judgments. The widespread reliance on these methods highlights their importance for the task and suggests that integrating multiple sources of evidence often outperforms any single model. One instructive finding, reported by *bipboopbipboop*, is that *where* fusion is applied matters: injecting a complementary signal into the candidate pool *before* the final listwise stage was considerably more effective than re-blending scores afterwards, because a hard top-n cutoff makes recall at the cut the binding constraint rather than the reranker's scoring ability.

4.3. The Role of Large Language Models

One of the most relevant differences with respect to the first edition of the task is the markedly more extensive use of LLMs. Whereas the previous edition provided only limited contextual information (job titles), the richer documents in this edition invited several teams to exploit generative models and prompting techniques to evaluate or reorder candidates.

LLMs as rerankers. *CYUT*, *DevoMatcher*, *classum*, and *bipboopbipboop* incorporate LLM-based reranking. Several adopt *listwise* reranking, in which the model receives multiple candidates simultaneously and produces a comparative ordering: *DevoMatcher* sends the top-50 calibrated candidates to a listwise prompt grounded in taxonomy-normalized evidence, while preserving the retrieval ranking as a fallback when the LLM output is incomplete. This contrasts with the *pointwise* strategy also explored

by *bipboopbipboop*, where each candidate is scored independently rather than comparing with others. *bipboopbipboop* additionally proposes a *tournament-style* variant, in which candidates are compared through successive rounds of pairwise competition [30], seeded from a geometric-mean fusion over the top-60. A recurring empirical lesson across these teams is that the gains of the final LLM stage are bounded by the composition and depth of its input candidate set, rather than by its raw scoring ability.

LLMs for data augmentation and query expansion. Beyond reranking, LLMs are used elsewhere in the pipeline, mainly as a data-augmentation mechanism. *VerbaNex* uses them for query expansion, *Skillberg.app* applies LLM-based corpus enrichment that augments each job title with a generated description, and *CYUT* incorporates Hypothetical Document Embeddings (HyDE) [31], generating hypothetical résumé profiles from job descriptions and, in a reverse variant, hypothetical job descriptions from résumés. These techniques aim to enrich the original input with additional context, and thereby improve the system's ability to retrieve relevant candidates. In addition, *bipboopbipboop* employs an LLM-as-a-judge approach to assess candidate adequacy during the retrieval stage, making it the only team to inject LLM knowledge directly into the retrieval signal rather than confining it to a downstream reranking stage.

LLMs for structured parsing. Two distinctive cases are *QTPride* and *classum*, which use LLM-based parsing as a step prior to retrieval. *QTPride* converts unstructured résumés and vacancies into structured JSON representations and builds multiple retrieval views (full text, work experience, education, explicit skills, and ESCO skill graph) over them, while *classum* constructs enriched structured views that expose role compatibility, activity overlap, seniority, domain, and skill evidence before scoring. In both systems, the LLM serves to transform noisy free text into a representation on which retrieval becomes more reliable.

4.4. Taxonomy- and Graph-Based Knowledge

A further axis of differentiation is the use of structured occupational knowledge, most often the ESCO taxonomy [32], to inject domain structure that raw text overlap cannot capture. *QTPride* combines LLM-extracted entities with the ESCO skill graph, which it encodes with a Relational Graph Convolutional Network to learn structure-aware embeddings [33], thereby incorporating relational information between entities into retrieval. *DevoMatcher* uses ESCO as a *calibration* signal rather than a hard filter: extracted and mined skill phrases are mapped to ESCO URIs, embedded with Node2Vec, and used to adjust —but not override— the retrieval score, with a development-weighted graph that down-weights hub nodes via inverse document frequency. *Skillberg.app*, finally, grounds its entire approach in a proprietary cross-framework occupation/skill knowledge graph spanning several European and international frameworks, from which it derives the contrastive training pairs used to fine-tune its encoder. Across these systems, the consistent finding is that taxonomic structure is most useful as soft, calibrating evidence that links different phrasings of related skills, and least useful when applied as a hard exclusion rule.

4.5. Training and Fine-Tuning Strategies

From a training perspective, not all teams perform task-specific fine-tuning of their retrieval models; several rely entirely on strong off-the-shelf components, a choice partly dictated by the absence of training labels. However, where fine-tuning is applied, the strategies are varied and often sophisticated. *bipboopbipboop* adapts its dense encoders with LoRA [34] and a cosine-similarity objective. *Skillberg.app* proposes a six-stage curriculum-learning methodology that progressively broadens the training distribution from in-framework synonyms to cross-framework and multilingual contrasts, using `GISTEmbedLoss` for guided in-batch negative selection and `MatryoshkaLoss` to keep embeddings informative at reduced dimensionalities [35, 36, 37]. *HR_NLP* adopts a teacher–student knowledge-distillation strategy combined with iterative hard-negative mining, explicitly aiming to separate résumés

that share skills but differ in seniority. Both *Skillberg.app* and *HR_NLP* thus treat the discrimination of seniority and experience level —rather than topical similarity alone— as the central modeling difficulty, echoing an error pattern (related-domain and seniority false positives) noted independently by several teams. Finally, *UBCS* employs encoder–decoder models to perform query expansion over the development and test sets, reinforcing the broader observation that text generation is used in this edition not only for ranking but also to enrich query representations.

4.6. Handling of Gender Bias

Because Task A includes an explicit bias-controlled evaluation (Section 2.4), it is informative to note how the participating teams addressed it. The majority of systems did not introduce a dedicated bias-mitigation mechanism and instead relied on the gender-neutrality of their underlying representations. A notable exception is *DevoMatcher*, which applies *demographic masking* as an explicit procedural safeguard: contact cues, pronouns, honorifics, and similar surface markers are masked from all scoring text —used consistently by the lexical retriever, the embedding rankers, the skill-extraction components, and the LLM reranker— and the LLM prompt is instructed to disregard demographic details and reward only job-relevant evidence. The team is careful to characterize the resulting high RBO as a ranking-*stability* signal rather than a proof of fairness, observing that indirect demographic proxies can still persist in work-history text. This distinction between measurable ranking stability and substantive fairness is one of the more important conceptual takeaways of the edition's bias track.

5. Results

In this section, we report the official Task A results along the three evaluation settings introduced in Section 2.4: the multilingual quality ranking measured by MAP, the cross-lingual English-to-Spanish setting, and the bias-controlled analysis based on RBO. Throughout all the results tables, we keep one row per team, retaining each team's best official test submission, and we include the organizer baseline for reference.

5.1. Multilingual Ranking Performance

Table 3 reports MAP scores across the monolingual English–English and Spanish–Spanish scenarios, and their average. *classum* achieved the best overall performance, with an average MAP of 0.7140, and led both languages individually (0.7122 on en–en and 0.7157 on es–es). *QTPride* ranked second overall, also obtaining the second-best result in each language, and *bipboopbipboop* completed the top three with closely balanced English and Spanish scores. Beyond the podium, *Skillberg.app* is notable for a strong English score —close to the second-best result in that language— paired with a lower Spanish score (0.6384), while *CYUT* shows the opposite asymmetry, performing better in Spanish than in English. Every participating team that produced valid test submissions surpassed the organizer baseline; the spread among participants' scores, however —roughly 0.22 MAP points from top to bottom— indicates that the methodological choices analyzed in Section 4 translate into substantial differences in ranking quality.

Table 3: Overview of team results for Task A, monolingual setting. Average MAP is the mean of MAP(en–en) and MAP(es–es). Best value per column in bold, second best underlined.

Team	System ID	Avg	MAP(en–en)	MAP(es–es)
classum	715941	**0.7140**	**0.7122**	**0.7157**
QTPride	696980	<u>0.6632</u>	<u>0.6622</u>	<u>0.6641</u>
bipboopbipboop	715693	0.6532	0.6552	0.6512
Skillberg.app	709705	0.6496	0.6608	0.6384
ui-nlp	715491	0.6264	0.6327	0.6202
CYUT	702458	0.6094	0.5974	0.6213
DevoMatcher	999999	0.5989	0.5966	0.6012
HR_NLP	715595	0.5605	0.5540	0.5669
VerbaNex	710446	0.5576	0.5614	0.5538
qwerity	699840	0.5066	0.5516	0.4616
UBCS	701328	0.4905	0.5144	0.4666
TalentCLEF Baseline	690133	0.3828	0.4001	0.3655

The two top-performing systems share a defining characteristic: both transform job descriptions and résumés into more comparable semantic representations before ranking, rather than scoring raw text directly. *classum* relied on semantic enrichment of query and corpus elements, including extracted skill and task profiles compared against structured representations of responsibilities, seniority, domain, and role compatibility inferred from the development set; these were then combined through multiple dense representations, LLM-based listwise and rubric scoring, score fusion, and reranking. *QTPride* followed a related multi-view strategy, first parsing each document into a structured JSON representation with an LLM and then fusing full-text, work-experience, and ESCO knowledge-graph retrieval views through reciprocal rank fusion. The same emphasis on structure appears in *Skillberg.app*, which incorporated a proprietary job–skill knowledge graph into its training. The third-ranked *bipboopbipboop* instead invested in a multi-stage reranking pipeline: candidate résumés were scored by complementary matchers (neural rerankers, bi-encoders, and an LLM-as-a-judge), fused by geometric-mean rank fusion, and refined with an LLM-based listwise tournament over the top candidates.

Taken together, these results suggest that contextualized job–person matching benefits from going beyond plain-text similarity. Structured document understanding, extracted entities, graph-derived relations, evidence fusion, and selective reranking each capture a different dimension of candidate–job alignment, and the systems that combined several of these mechanisms occupied the top of the leaderboard. This is consistent with the qualitative error patterns reported by participants in Section 4.

5.2. Cross-Lingual Performance

The cross-lingual setting evaluates MAP for the English–Spanish pair, in which English queries are matched against a Spanish corpus. Results are shown in Table 4.

Table 4: Overview of team results for Task A, cross-lingual setting (English query, Spanish corpus). Best value per column in bold, second best underlined.

Team	System ID	MAP(en–es)
classum	707960	**0.7044**
bipboopbipboop	715693	<u>0.6438</u>
Skillberg.app	690318	0.6373
QTPride	696980	0.6355
ui-nlp	715491	0.5995
CYUT	702388	0.5851
HR_NLP	715595	0.5518
DevoMatcher	999999	0.5293
VerbaNex	703373	0.4613
TalentCLEF Baseline	690133	0.3562
UBCS	701251	0.3562

The strongest cross-lingual systems reproduced the patterns observed in the monolingual setting. The best-performing system, by *classum*, combined multiple cross-lingual dense representations with its enriched job and résumé views. *bipboopbipboop* adapted its pipeline to the cross-lingual condition by retaining only two of its four matchers —a multilingual BGE-M3 bi-encoder and a neural reranker— before applying the LLM-based listwise tournament; here the BGE-M3 component plausibly supplied most of the multilingual semantic signal entering the final fused ranking. The most informative observation is that, for the teams at the top of the leaderboard, the gap between cross-lingual and monolingual performance was small. This near-parity suggests that the leading architectures are genuinely language-agnostic rather than English-centric, and it contrasts with the experience of teams that handled the cross-lingual case through a translate-to-English anchor (Section 4.1), several of which reported their weakest results in this direction. Cross-lingual matching, long regarded as the hardest of the three settings, thus appears largely solved by the strongest multilingual encoders.

5.3. Bias Evaluation

Beyond standard retrieval effectiveness, we conducted a bias-oriented analysis measuring the consistency of rankings across gendered variants of the same records. The synthetic evaluation data was built from paired profiles in which only gender-marked elements —names and words with either lexical or morphological gender referring to the candidate— were modified, while all other information was held constant. This isolates the effect of the perturbation: a system that produces nearly identical rankings for the masculine and feminine variants can be considered robust to it, whereas larger discrepancies signal sensitivity to gender-marked surface cues. As in previous works [38], we quantify this consistency with RBO comparing the gendered variants in the English–English, English–Spanish, and Spanish–Spanish scenarios. Per-variant and average values are reported in Table 5. *classum* achieved the highest overall RBO, leading every direction. *HR_NLP* ranked second overall, obtaining the second-best score in every setting. RBO and MAP are not interchangeable: the bias table re-ranks the submissions, and a team with mid-table MAP now ranks among the most stable, indicating that ranking robustness to gender perturbations is partly decoupled from overall retrieval accuracy.

Table 5: Overview of team results for Task A using RBO. Higher values indicate greater ranking stability under the gender perturbation. Best value per column in bold, second best underlined.

Team	System ID	Avg	RBO(en–en)	RBO(en–es)	RBO(es–es)
classum	707960	**0.9904**	**0.9931**	**0.9892**	**0.9890**
HR_NLP	715595	<u>0.9521</u>	<u>0.9558</u>	<u>0.9466</u>	<u>0.9538</u>
Skillberg.app	709705	0.9338	0.9369	0.9309	0.9336
DevoMatcher	999999	0.9091	0.9492	0.8667	0.9114
QTPride	696980	0.8842	0.8971	0.8575	0.8979
bipboopbipboop	715693	0.8727	0.8714	0.8631	0.8837
CYUT	702388	0.8581	0.8528	0.8421	0.8795
UBCS	701328	0.8211	0.8624	0.7665	0.8344
ui-nlp	715491	0.8110	0.8251	0.7789	0.8291
TalentCLEF Baseline	690133	0.7854	0.7919	0.7800	0.7843
VerbaNex	709530	0.6960	0.7580	0.5848	0.7453

The very high scores obtained by *classum* indicate that its rankings were almost unaffected by the gender perturbations. A plausible explanation is that extracting task- and skill-oriented profiles reduces the influence of surface-level lexical changes, while the aggregation of multiple dense embedding models, LLM-based scoring, score fusion, and reranking further stabilizes the ranking by drawing on complementary signals rather than a single text representation. The bias track also exposes a wide range of behavior: scores span from above 0.99 to below 0.70, and the cross-lingual direction is consistently the least stable column for most teams, suggesting that gender robustness and cross-lingual alignment interact. These results should nonetheless be read with care. As discussed in Section 4.6, none of the top systems reports a dedicated bias-mitigation component, and the pretrained models on which they rely may still inherit biases from their training data. *DevoMatcher* is the only team to apply an explicit safeguard (demographic masking), and even there the authors frame the outcome as ranking stability rather than fairness.

5.4. Meta-Analysis: The Headroom of Per-Query System Fusion

The per-team results in Section 5 establish how individual systems ranked, but leave open a central question: how much of the remaining error is idiosyncratic to individual systems, and how much could be recovered by *combining* system outputs? To probe this, we conducted a post-hoc *per-query oracle fusion*. For each of the three language pairs and each of the 40 evaluation queries, the oracle searches a grid of fusion combinations —fusion method together with a weighted subset of the top-6 candidate systems— and retains, for that query alone, the combination that maximizes MAP. The candidate pool is identical across language pairs and comprises the six strongest participants: *classum*, *CYUT*, *bipboopbipboop*, *Skillberg.app*, *QTPride*, and *ui-nlp*.

Obtained numbers must be read as evidence about *where headroom lies* and *which methodological choices co-occur with strong fusions*, not as controlled ablations.

5.4.1. Per-Query Adaptivity Is the Dominant Source of Headroom

Table 6 contrasts three reference points per language pair, all scored by MAP over the same set of covered queries. The *best single* system is the strongest individual run in the pool, with no fusion applied. The *global* oracle searches the full grid for the single combination [4] that maximizes the *mean* MAP across all queries, and then applies that same combination to every query; it is therefore the best fixed ensemble one could commit to without query-specific tuning. Figure 2 makes the global

[4] A fusion method together with one fixed weight vector over a subset of the systems.

oracle concrete, listing for each monolingual pair the single weighted-sum combination. The *per-query* oracle relaxes this constraint, letting the winning combination be chosen independently for each query, and reports the mean of those per-query maxima; it is an upper bound attainable only with oracular knowledge of which combination wins on each query. Because the per-query oracle is free to reuse the global combination on every query, it can never fall below the global oracle, and the gap between the two isolates the headroom due purely to query-level adaptation. Empirically, the global oracle barely improves on the best single system, whereas the per-query oracle lifts MAP substantially. Almost none of the available headroom comes from finding a better *fixed* ensemble; nearly all of it comes from *adapting the combination to each query*. This reframes the leaderboard: the gap between the best participant and the per-query ceiling is real, but it is unlocked only by query-level routing, which no participating system attempted.

Table 6: Oracle fusion ceiling by language pair (MAP scores). "Best single" is the strongest individual system in the pool, with no fusion. The *global* oracle is the single fusion combination (method and one weight vector over a subset of systems) that maximizes mean MAP across queries, applied identically to every query. The *per-query* oracle adapts the combination independently for each query (mean of the per-query maxima).

Lang. pair	Best single	Global oracle	Per-query oracle
en–en	0.712	0.715	0.781
es–es	0.715	0.728	0.784
en–es	0.705	0.726	0.784

```
English-English
  X* = [sum] sisobus=1, skillberg=0.5, bipboopbipboop=1, johanan=1

Spanish-Spanish
  X* = [sum] sisobus=1, bipboopbipboop=0.5, skillberg=0.5, ibtihalql=0.5
```

Figure 2: Global-oracle fusion combinations $X^\star$ for the monolingual pairs: the single fixed weighted-sum (`[sum]`) combination that maximizes mean MAP across all queries, together with the weight the oracle assigns to each system. System names are the participants' run identifiers.

5.4.2. What the Winning Fusions Are Made Of

The oracle does not draw evenly from the pool. Table 7 reports how often each team is selected, and the mean MAP of the combinations in which it appears. *classum* is selected most often (78 times) and contributes the most weight, confirming that the strongest single system is also the most broadly useful fusion ingredient. Yet selection frequency and combination quality are not perfectly aligned: *ui-nlp*, selected least often (41 times), appears in the highest-MAP combinations on average, while *CYUT* pairs a high selection count with a comparably high mean. Some systems are valuable not because they are strong everywhere but because they are *complementary*, contributing decisive signal on the specific queries where they are chosen.

Table 7: Team participation in the per-query oracle fusions (across all language pairs and queries). "Selected" counts oracle combinations including the team; "Mean MAP" is over the combinations in which the team appears.

Team	# Selected	Mean MAP
classum	78	0.789
CYUT	60	0.791
bipboopbipboop	54	0.761
Skillberg.app	45	0.777
QTPride	45	0.737
ui-nlp	41	0.813

The informative observations come from features that vary across teams, summarized in Table 8. The clearest positive association is with a decoder component, which co-occurs with a +0.158 MAP difference and appears in 98% of winning combinations. Generative reranking and parsing are characteristic of the systems the oracle prefers. This aligns with the leaderboard, where all three top systems use LLMs substantively (Section 4.3). More surprising is the negative association of `Graphs/KG` (−0.047): the two graph-based systems (*Skillberg.app*, *QTPride*) are strong individually, but their structured signal appears partially redundant with what the dense and generative systems already capture, so combinations *without* a graph component scored higher on average within this pool.

Table 8: Feature impact across the 120 per-query winning combinations, restricted to features that vary across the pool. "Presence" is the fraction of winning combinations containing at least one system with the feature; ΔMAP is the mean MAP of combinations with the feature minus those without. Universal features (Encoder, Retrieval, Reranking, Rank Fusion, Prompt Engineering) are omitted as non-discriminative.

Feature	Presence	MAP (with / w/o)	ΔMAP
Decoder	98%	0.787 / 0.629	+0.158
External Data	50%	0.791 / 0.775	+0.016
Contrastive Learning	38%	0.777 / 0.786	−0.009
Custom Loss	38%	0.777 / 0.786	−0.009
Graphs/KG	64%	0.766 / 0.813	−0.047

6. Conclusions

This paper presented an overview of Task A of TalentCLEF 2026, which we reformulated from the job title matching benchmark of the previous edition into a highly relevant, realistic problem of contextualized job–person matching: given a job vacancy, systems rank a pool of candidate résumés by their suitability. To support this, we released a multilingual synthetic dataset (English and Spanish) and evaluated submissions under three complementary settings. The clear real-world relevance of the task has attracted significant interest, with 91 registered participants and 21 teams submitting runs from both academia and industry. Eleven of these teams contributed working notes, allowing us to analyze a rich diversity of approaches along five recurring axes (multi-stage retrieval, reranking and rank fusion, the use of LLMs, taxonomy- and graph-based knowledge, and training strategies). We complemented the official results with a post-hoc meta-analysis of per-query system fusion. Ultimately, the variety of solutions submitted and the remaining performance gaps underscore that while substantial progress has been made, contextualized talent matching remains an open challenge; the problem is not yet fully

solved, highlighting the clear need for continued research in this space.

Building upon the success of this year's edition, several compelling directions emerge for future iterations of TalentCLEF. First, we propose a novel upskilling and reskilling recommendation task, where systems must automatically identify specific training modules or educational paths from a predefined catalog that a candidate should complete to bridge their competency gaps. This task would operate under realistic, imperfect hiring conditions where candidate pools contain *partially matching* profiles, shifting the focus from binary filtering to active professional development. Second, in alignment with current trends in explainable AI (XAI) and legislative frameworks for algorithmic accountability in recruitment, future tasks should transcend raw ranking indices by requiring systems to generate natural language justifications or reports detailing the underlying rationale for a matched or non-matched decision. Third, we envision extending the matching paradigm from individual evaluation to holistic team-building; once a primary match is identified for a vacancy, the systems would be tasked with ranking subsequent candidates based on how well they complement the existing team composition and fill collective operational voids. Finally, future editions could introduce temporal and semantic structure challenges, such as modeling job-to-job mobility to predict a candidate's next logical career transition

Declaration on Generative AI

During the preparation of this work, the authors used Opus 4.8 to check grammar and spelling. After using this tool, the authors reviewed and edited the content as needed and assume full responsibility for the content of the publication.

References

[1] World Economic Forum, The Reskilling Revolution: 350 Million People Reached with Future-Ready Skills, Education and Jobs, https://www.weforum.org/press/2023/01/the-reskilling-revolution-350-million-people-reached-with-future-ready-skills-education-and-jobs/, 2023. Press release.

[2] OECD, Empowering the workforce in the context of a skills-first approach, 2025. URL: https://www.oecd.org/en/publications/empowering-the-workforce-in-the-context-of-a-skills-first-approach_345b6528-en.html, Accessed: 2026-06-03.

[3] LinkedIn, Work Change Report: AI Is Coming to Work, 2025. URL: https://economicgraph.linkedin.com/content/dam/me/economicgraph/en-us/PDF/Work-Change-Report.pdf, Accessed: 2025-05-18.

[4] K. Khelkhal, D. Lanasri, Smart-Hiring: An Explainable end-to-end Pipeline for CV Information Extraction and Job Matching, arXiv preprint arXiv:2511.02537 (2025).

[5] L. Gasco, H. Fabregat, L. García-Sardiña, P. Estrella, D. Deniz, A. Rodrigo, R. Zbib, Overview of the TalentCLEF 2025: Skill and Job Title Intelligence for Human Capital Management, in: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2025, pp. 464–485.

[6] L. Gasco, H. Fabregat, L. García-Sardiña, D. Deniz, A. Rodrigo, P. Estrella, R. Zbib, TalentCLEF at CLEF2025: Skill and Job Title Intelligence for Human Capital Management, in: European Conference on Information Retrieval, Springer, 2025, pp. 479–486.

[7] W. Veys, L. Gasco, M. De Lange, J.-J. Decorte, Overview of TalentCLEF 2026: Task B — Job-Skill Matching with Skill Type Classification, in: CLEF (Working Notes), 2026.

[8] R. Zbib, L. L. Alvarez, F. Retyk, R. Poves, J. Aizpuru, H. Fabregat, V. Simkus, E. G. Casademont, Learning Job Titles Similarity from Noisy Skill Labels, CoRR abs/2207.00494 (2022). URL: https://doi.org/10.48550/arXiv.2207.00494. doi:10.48550/ARXIV.2207.00494. arXiv:2207.00494.

[9] D. Deniz, F. Retyk, L. García-Sardiña, H. Fabregat, L. Gascó, R. Zbib, Combined Unsupervised and Contrastive Learning for Multilingual Job Recommendation, in: M. Kaya, T. Bogers, D. Graus, C. Johnson, J. Decorte, T. D. Bie (Eds.), Proceedings of the 4th Workshop on Recommender Systems

for Human Resources (RecSys-in-HR 2024) co-located with the 18th ACM Conference on Recommender Systems (RecSys 2024), Bari, Italy, 14th-18th October 2024, volume 3788 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2024. URL: https://ceur-ws.org/Vol-3788/RecSysHR2024-paper_3.pdf.

[10] L. García-Sardiña, F. Retyk, H. Fabregat, L. Alvarez Lacasa, R. Poves, R. Zbib, Normalisation of Education Information in Digitalised Recruitment Processes, Procesamiento del Lenguaje Natural 71 (2023) 63–73.

[11] S. Vaishampayan, H. Leary, Y. B. Alebachew, L. Hickman, B. Stevenor, W. Beck, C. Brown, Human and llm-based resume matching: An observational study, in: Findings of the Association for Computational Linguistics: NAACL 2025, 2025, pp. 4823–4838.

[12] P. Ghosh, V. Sadaphal, Jobrecogpt–explainable job recommendations using llms, arXiv preprint arXiv:2309.11805 (2023).

[13] F. P.-W. Lo, J. Qiu, Z. Wang, H. Yu, Y. Chen, G. Zhang, B. Lo, Ai hiring with llms: A context-aware and explainable multi-agent framework for resume screening, in: Proceedings of the computer vision and pattern recognition conference, 2025, pp. 4184–4193.

[14] L. Gasco, H. Fabregat, L. García-Sardiña, P. Estrella, C. P. Carrino, D. Deniz, A. Rodrigo, R. Zbib, TalentCLEF at CLEF2026: Skill and Job Title Intelligence for Human Capital Management, in: European Conference on Information Retrieval, Springer, 2026.

[15] L. Gasco, H. Fabregat, L. García-Sardiña, P. Estrella, W. Veys, C. P. Carrino, M. De Lange, D. Deniz Cerpa, A. Rodrigo, J.-J. Decorte, R. Zbib, Overview of the talentclef 2026: Skill and job title intelligence for human capital management, in: International Conference of the Cross-Language Evaluation Forum for European Languages, 2026.

[16] A. Palomino, Hybrid Dense-Sparse Retrieval and Re-Ranking for Multilingual Resume Retrieval from Job Descriptions, in: CLEF (Working Notes), 2026.

[17] J. Poerwanto, S.-H. Wu, CYUT at TalentCLEF 2026: Zero-Shot Multi-View Retrieval for Job-Person Matching, in: CLEF (Working Notes), 2026.

[18] S. Kim, M. Choi, S. Lee, Semantic Enrichment for Resume Ranking and Skill Retrieval at TalentCLEF 2026, in: CLEF (Working Notes), 2026.

[19] A. Riabi, M. Essayeh, Taxonomy-Aware Hybrid Retrieval with Bounded LLM Reranking for TalentCLEF 2026 Task A, in: CLEF (Working Notes), 2026.

[20] M. Vielvoye, Skillberg at TalentCLEF 2026: Cross-Framework Knowledge-Graph Grounded Multilingual Encoders for Job–Person and Job–Skill Matching, in: CLEF (Working Notes), 2026.

[21] D.-A. Dumitru, Ranking job-resume similarity using cross-lingual bi-encoding, in: CLEF (Working Notes), 2026.

[22] E. Thuma, G. Anderson, A Union Fusion of PyTerrier, Doc2Query, BM25 and Multilingual Embeddings for Job Title Matching, in: CLEF (Working Notes), 2026.

[23] M. Moreno, J. C. Martinez-Santos, E. Puertas, VerbaNex at TalentCLEF 2026: A Hybrid Multilingual Retrieval and Approach for Contextualized Job-Person Matching, in: CLEF (Working Notes), 2026.

[24] I. Q. Luthfiyyah, S. Yudhoatmojo, I. Budi, ui-nlp at TalentCLEF 2026: Multilingual Bi-Encoder Retrieval with Ensemble Strategy for Job-Person and Job-Skill Matching, in: CLEF (Working Notes), 2026.

[25] L. Hemamou, R. Dupont, Retrieve, Rerank, Survive: LLM-Listwise Pipelines for Job–Person and Job–Skill Matching at TalentCLEF 2026, in: CLEF (Working Notes), 2026.

[26] H.-H.-H. Tran, QTPride @TalentCLEF 2026: Towards Multi-View Job-Person Matching with LLM Parsing and Knowledge Graph-Enhanced Retrieval, in: CLEF (Working Notes), 2026.

[27] J. Decorte, J. V. Hautte, T. Demeester, C. Develder, JobBERT: Understanding Job Titles through Skills, CoRR abs/2109.09605 (2021). URL: https://arxiv.org/abs/2109.09605. arXiv:2109.09605.

[28] C. Lassance, H. Déjean, T. Formal, S. Clinchant, SPLADE-v3: New baselines for SPLADE, arXiv preprint arXiv:2403.06789 (2024).

[29] O. Khattab, M. Zaharia, Colbert: Efficient and effective passage search via contextualized late interaction over bert, in: Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, 2020, pp. 39–48.

[30] Y. Chen, Q. Liu, Y. Zhang, W. Sun, X. Ma, W. Yang, D. Shi, J. Mao, D. Yin, Tourrank: Utilizing large

language models for documents ranking with a tournament-inspired strategy, in: Proceedings of the ACM on Web Conference 2025, 2025, pp. 1638–1652.

[31] M. Li, X. Lv, J. Zou, T. Chen, C. Zhang, S. An, E. Nie, G. Zhou, Query Expansion in the Age of Pre-trained and Large Language Models: A Comprehensive Survey, ACM Transactions on Information Systems (2025).

[32] M. le Vrang, A. Papantoniou, E. Pauwels, P. Fannes, D. Vandensteen, J. D. Smedt, ESCO: Boosting Job Matching in Europe with Semantic Interoperability, Computer 47 (2014) 57–64. URL: https://doi.org/10.1109/MC.2014.283. doi:10.1109/MC.2014.283.

[33] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: European semantic web conference, Springer, 2018, pp. 593–607.

[34] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al., Lora: Low-rank adaptation of large language models., Iclr 1 (2022) 3.

[35] X. Wang, Y. Chen, W. Zhu, A survey on curriculum learning, IEEE transactions on pattern analysis and machine intelligence 44 (2021) 4555–4576.

[36] A. V. Solatorio, Gistembed: Guided in-sample selection of training negatives for text embedding fine-tuning, arXiv preprint arXiv:2402.16829 (2024).

[37] A. Kusupati, G. Bhatt, A. Rege, M. Wallingford, A. Sinha, V. Ramanujan, W. Howard-Snyder, K. Chen, S. Kakade, P. Jain, et al., Matryoshka representation learning, Advances in Neural Information Processing Systems 35 (2022) 30233–30249.

[38] L. García-Sardiña, H. Fabregat, D. Deniz, R. Zbib, Measuring Gender Bias in Job Title Matching for Grammatical Gender Languages, CoRR abs/2509.13803 (2025). URL: https://doi.org/10.48550/arXiv.2509.13803. doi:10.48550/ARXIV.2509.13803. arXiv:2509.13803.