# From Retrieval to Ranking: A Two-Stage Neural Framework for Automated Skill Extraction

Aleksander Bielinski[1,†], David Brazier[1]

[1]Edinburgh Napier University, School of Computing, Engineering & The Built Environment

**Abstract**

Automated skill extraction from job postings is crucial for understanding labour market dynamics, but current approaches struggle to balance retrieval efficiency with ranking accuracy. Most existing methods focus on either dense retrieval for candidate generation or multi-label classification, failing to leverage the complementary strengths of both paradigms. While recent work has begun exploring retrieve-and-rank pipelines for skill extraction using Large Language Models (LLMs) for ranking, we propose training dedicated neural models for both retrieval and ranking stages. In our two-stage approach, the bi-encoder efficiently retrieves skill candidates, while the cross-encoder provides precise ranking using focal loss optimisation. We evaluate both stages separately on publicly available datasets. Our bi-encoder achieves up to 4.78 percentage points improvement in RP@5 over existing baselines, while our cross-encoder demonstrates up to 30.54 percentage points improvement in micro-F1 compared to LLM-based ranking methods. Additionally, our bi-encoder shows strong zero-shot performance on held-out skills. The framework leverages public datasets and freely available skill taxonomies like ESCO, promoting scalable and reproducible skill extraction. We release our code and configurations to encourage further research, available at https://github.com/AleksanderB-hub/Multi-Stage-Pipeline-Skill-Extraction.

**Keywords**

Skill Extraction, Multi-Stage Information Retrieval, Contrastive Learning, ESCO

## 1. Introduction

Fuelled by technological developments and societal changes, today's labour market transforms dynamically, making the assessment of job market demand an increasingly challenging task [1]. With the European Skills, Competences, Qualifications and Occupations (ESCO) taxonomy alone containing nearly 14,000 distinct skills, and millions of job postings published daily across various platforms, the need for automated skill extraction has never been more critical. Skill extraction plays a pivotal role in this task as it allows for the extraction of competencies from available data (e.g., resumes, job postings) and mapping them to a standardised taxonomy. This enables HR professionals and policymakers to better understand current market trends and support workforce planning, ensuring the efficient functioning of labour markets. The growing importance of such systems is evidenced by the recent surge in research on automated skill extraction [2, 3].

The task of skill extraction from job postings presents unique challenges that distinguish it from traditional text classification. First, skills are often mentioned implicitly rather than explicitly [4]. This implicit nature renders simple keyword matching approaches ineffective. Second, the diverse vocabulary used across industries and regions means that even accurately extracted skills must be normalised to a standardised taxonomy like ESCO to enable meaningful analysis and comparison across markets and time periods.

The main problem with developing skill extraction systems is the scarcity of real-life annotation data [5]. This was partially addressed by the creation of artificially generated job posting data, which is later used to train the models for automated skill extraction. However, these artificial datasets often fail to capture the full complexity and linguistic variety of real-world job postings. With this creating a potential gap between training and deployment scenarios, it is necessary to carefully balance the use of synthetic training data with limited real-world resources.

Beyond data availability, existing approaches to skill extraction face architectural limitations that constrain their effectiveness. Current approaches either prioritise the direct skill classification or relevant candidate retrieval (dense retrieval), where each query (job description sentence) is provided with a list of relevant documents (matching skills). The issue with the standard classification is that it is limited to the data it was trained on, consequently impairing the generalisability of such solutions. This is particularly problematic due to the constantly evolving skill space. Conversely, dense retrieval approaches frame skill extraction as a similarity search problem. By searching for similar skills from the entire taxonomy, they often return numerous irrelevant candidates, making accurate skill profile extraction challenging.

Recent advances in Information Retrieval (IR) suggest that combining dense retrieval with ranking capability offers significant improvements over single-stage systems [6, 7]. These architectures combine a dense retriever (e.g., bi-encoder) with a ranking model (e.g., cross-encoder), leveraging the complementary strengths of these two methods. Given that skill extraction requires the retrieval of relevant skills from large taxonomies, these two-stage architectures present a natural fit. However, while Clavié and Soulié [8] and D'Oosterlinck et al. [9] recently showed that Large Language Models (LLMs) rankers could improve skill extraction, the potential of training dedicated neural architectures for both stages remains unexplored. This represents a significant gap, as purpose-built rankers can offer better performance and efficiency than general-purpose LLMs.

We propose a novel two-stage neural architecture that adapts successful IR practices to the unique requirements of skill extraction. Our approach combines a bi-encoder for efficient candidate retrieval from large skill taxonomies with a cross-encoder for precise skill identification. In the first stage, we fine-tune a bi-encoder using a curriculum learning strategy that leverages freely available ESCO skill definitions. The model first learns to associate skills with their

*RecSys in HR'25: The 5th Workshop on Recommender Systems for Human Resources, in conjunction with the 19th ACM Conference on Recommender Systems, September 22–26, 2025, Prague, Czech Republic.*

†Corresponding author.

a.bielinski@napier.ac.uk (A. Bielinski); d.brazier@napier.ac.uk (D. Brazier)

0000-0002-7254-1290 (A. Bielinski); 0000-0001-9225-6174 (D. Brazier)

canonical definitions before training on synthetic job posting data. This approach not only improves performance on seen skills but also enables strong zero-shot generalisation to skills excluded from training. The bi-encoder retrieves the top-K most relevant skills for each job description sentence based on embedding similarity.

In the second stage, we employ a cross-encoder that ranks the retrieved candidates using a binary classification objective. Unlike traditional ranking approaches that reorder candidates, our cross-encoder makes explicit decisions about whether each skill is truly relevant to the job description. Such a design choice aligns with the multi-label nature of skill extraction, where a given query might describe multiple or no skills at all. This approach leverages a cross-encoder's ability to jointly process query and documents, allowing the capture of subtle semantic relationships that independent encoding might miss. Our multi-stage approach is visualised in Figure 1.

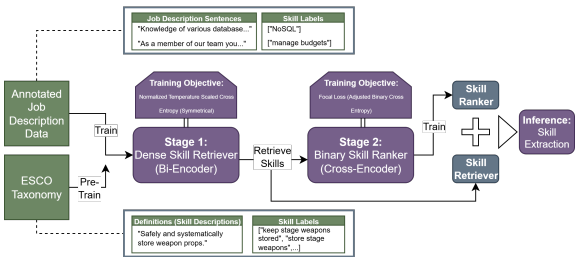


**Figure 1:** General overview of the proposed retrieve-and-rank method for skill extraction. In Stage 1, a dense bi-encoder is trained on annotated job description data following the pre-training using the ESCO taxonomy; this forms our curriculum strategy. Stage 2 uses the candidate skill labels retrieved in Stage 1 to train the model for binary ranking of skills. At inference, the candidate predictions from Stage 1 are provided to the Stage 2 model (ranker) to predict skill labels.

Comprehensive evaluation across established skill extraction benchmarks demonstrates the effectiveness of this two-stage approach. The bi-encoder achieves up to 4.78 percentage points improvement in R-Precision@5 (RP@5) compared to existing dense retrieval baselines while maintaining strong retrieval performance on held-out skills in zero-shot settings. When combined with the cross-encoder, our complete pipeline achieves F1 scores up to 30.54 percentage points higher than LLM-based ranking methods. These results validate that carefully designed two-stage neural architectures can significantly improve skill extraction while maintaining the efficiency required for practical deployment.

**Contributions.** In summary, our main contributions are:

- A curriculum-trained dense retriever over taxonomy labels for candidate skill generation, showcasing strong zero-shot retrieval capabilities.
- A task-specific, supervised cross-encoder ranker for multi-label skill extraction, delivering strong classification performance across public benchmarks.

# 2. Relevant Work

## 2.1. Skill Extraction

Early approaches to skill extraction were mostly limited to span-level extraction. The task consisted of retrieval of relevant fragments from the sentences (job descriptions or resumes) to train Named Entity Recognition (NER) models [10, 11]. Seeing the rapid advances in LLMs, researchers demonstrated how generative AI can be leveraged for a span-level skill extraction [12, 13]. Some notable work also exists using the graph neural networks for context-aware skill extraction [14, 15]. Despite their strong performance, the main issue with such approaches is the lack of skill label normalisation. The retrieved spans are not linked to the standardised taxonomy (e.g., ESCO), making these techniques less applicable in real-world scenarios. To address that, authors in [16] demonstrate how the identification of relevant skill spans can aid downstream classification of competencies, highlighting the complementary nature of such approaches. Similar techniques were later expanded in [17], where job descriptions are directly matched with skills in a taxonomy. Another issue lies in reliance on high-quality annotation data, which is costly to obtain, especially when considering the necessary involvement of human resource domain experts [12, 2].

Large-scale skill and occupation taxonomies offer a breadth of potentially useful information for skill extraction tasks, such as co-dependency of competencies, hierarchical classification, etc. Building on this, researchers in [18] explored the use of ESCO skill labels as weak supervision signals. Furthermore, work by Decorte et al. [19] showcased that taxonomy-based weak supervision signals can be combined with classification models to satisfy skill normalisation of extracted competencies. The role of such taxonomies in supporting the performance on downstream tasks has been further highlighted in [20], where information from ESCO was used as pre-training signals for a skill extraction model.

Understanding the importance of skill normalisation and challenges around sourcing annotation data, research shifted towards the generation of artificial job description data [2, 3]. Partially fuelled by the wealth of information offered by ESCO, these works showcased how incorporating definition (skill description) information into the generating pipeline increases the quality of the examples. In addition, [2] also shows that definitions can serve as a training signal on their own. Most recently, Decorte et al. [21] introduced a novel end-to-end skill extraction architecture, achieving strong results on skill retrieval benchmarks. Their work utilises the skill definitions as training signals, further confirming the benefits of incorporating taxonomy information into skill extraction pipelines.

In light of this evidence, our work introduces a novel pre-training phase utilising the skill descriptions and labels provided by the ESCO taxonomy. This builds on the reported success of curriculum learning in information retrieval (IR), where models benefit from training on progressively complex examples [22]. Such approaches have shown promise in dense retrieval tasks [23], where domain-specific pre-training improves downstream performance. However, unlike prior work that requires specialised architectures or training procedures, our curriculum learning strategy maintains the standard bi-encoder architecture while leveraging freely available taxonomy data.

## 2.2. Two-Stage Retrieval Architectures

Modern information retrieval has undergone a fundamental shift from sparse keyword matching to dense neural representations, revolutionising how systems retrieve and rank information. While traditional methods like BM25 remain competitive baselines, neural approaches, particularly bi- and cross-encoder architectures, have demonstrated superior performance across diverse IR tasks [24]. Nonetheless, these methods face an inherent trade-off. Bi-encoders enable efficient retrieval through pre-computed representations but sacrifice fine-grained query-document interaction, while cross-encoders that jointly process query-document pairs provide superior relevance modelling but cannot scale to large collections.

This challenge has given rise to two-stage retrieval architectures that combine dense retrievers with dedicated rankers to achieve superior retrieval quality. Early work utilised the BM25 for the retrieval stage and combined it with a BERT-based ranker in a question answering task [25]. However, modern systems increasingly employ neural methods in both stages. Such methods often use bi-encoders for efficient candidate retrieval followed by cross-encoders for precise ranking [26, 27]. While such neural two-stage systems have shown strong results, recent work has explored the use of LLM-based rankers across various domains [28, 29], including skill extraction [8, 9]. Despite recent interest in LLM-based rankers, bi-encoder/cross-encoder architectures remain widely deployed due to their predictable computational costs and proven effectiveness.

Building on the success of two-stage retrieval systems in IR, we adapt this paradigm to skill extraction. To the best of our knowledge, we propose the first architecture combining bi-encoder retrieval and cross-encoder ranking models specifically designed for this task. In contrast to LLM-based ranking for skills [8, 9], we train dedicated neural models for both stages, offering better efficiency and performance while leveraging the complementary strengths of both encoder types.

# 3. Methodology

We present a two-stage neural architecture for skill extraction that frames the task as dense retrieval followed by binary ranking. Our approach leverages curriculum learning to maximise the utility of limited training data, combining synthetic datasets with freely available ESCO skill definitions. This section details our problem formulation, data configuration strategy, and the design of both pipeline stages.

## 3.1. Problem Statement

In our case, we aim to extract all relevant skills for a given job description fragment. For example, given the sentence:

> *"Be able to lead and motivate people and have good communication skills."*

The goal is to extract skills such as *communication*, *lead others*, and *motivate others* from a larger taxonomy of possible skills (e.g., ESCO). We approach this problem in two stages.

**Stage 1: Dense Skill Retriever (bi-encoder retriever).** We first retrieve a small subset of relevant skills from a large skill taxonomy. This is done by encoding the job sentence and each skill into dense vectors using a trained bi-encoder, and computing their cosine similarity. The top-K most similar skills are returned as candidates.

**Stage 2: Binary Skill Ranker (cross-encoder ranker).** Next, we refine this list using a trained cross-encoder model that jointly reads the sentence and each candidate skill, and assigns a relevance score. Skills above a tuned relevance threshold are predicted as relevant (see Section 3.5 for threshold tuning details).

**Assumption.** At inference, the ranker sees only retrieved candidates; thus, its effectiveness depends on Stage 1 retrieval quality. To create a representative training sample, we inject the missing gold labels into the training data (no injection at test time; see Section 3.5 for details).

**Evaluation.** Retrieval is assessed using RP@K and MRR. Ranking performance is measured using micro-F1 across all sentence-skill pairs. The use of each metric is justified in Section 4.1.

## 3.2. Overview of Available Datasets

Our experiments leverage both synthetic and real-world datasets to address the data scarcity challenge in skill extraction. For synthetic data, we utilise two complementary resources. First, the *DECORTE* dataset [2], which contains 138,240 artificially generated examples covering nearly the entire ESCO taxonomy. Secondly, we use *SKILLSKAPE* dataset [3], comprising 8,940 multi-skill examples divided into train, val and test sets, where each sentence can describe up to nine different skills. While *DECORTE* provides broad coverage of the ESCO taxonomy with clear single-skill associations, *SKILLSKAPE* better reflects real-world complexity where multiple skills co-occur within job requirements.

When it comes to real-world data, we employ three manually annotated datasets: *HOUSE*, containing 663 job description sentences annotated with ESCO labels, split into val and test sets; *TECH*, featuring 796 fully annotated job ad sentences (val + test); and *TECHWOLF* with 588 annotated examples (test). All these datasets were originally sourced from [10] and later annotated in [19] (*HOUSE*, *TECH*) and in [2] (*TECHWOLF*) using ESCO labels. Additionally, we incorporate the skill labels, alongside their synonyms and definitions from ESCO v1.1.0 [30], serving as a valuable knowledge source for our curriculum learning approach (*ESCO-D*).

## 3.3. Stage-Specific Data Configuration

Given the limited availability of real-world annotated data and the different requirements of our two-stage architecture, we strategically partition the datasets described above to serve distinct roles in training and evaluation. Table 1 presents the complete data allocation, which we designed following three key principles: (1) maintaining strict separation between training and test data for fair evaluation, (2) maximising the use of real-world examples where they provide the most benefit and (3) balancing the use of synthetically generated data to ensure efficient learning.

For Stage 1, we use only one example per skill from *DECORTE* despite the availability of ten. This decision is

based on preliminary experiments showing no significant performance improvement when using additional examples, provided the augmentation strategy from [2] is employed (see Section 5). However, as highlighted in [3], real-life job descriptions often describe multiple skills within a single sentence. Consequently, *SKILLSKAPE* provides job ad fragments which are both longer and more complex than those of *DECORTE*, albeit offering inferior taxonomy coverage. Therefore, we decided to combine these two datasets in our training data. We hypothesise that such a configuration satisfies both taxonomy coverage (i.e., each skill in ESCO has at least one example) and provides a more informative learning signal for our model. To take advantage of existing taxonomies, we further expand our training data with definitions from *ESCO-D*, which were shown to provide a strong training signal in skill extraction tasks [2]. The main training phase is preceded by pre-training, where both definitions and skill labels from *ESCO-D* are used, forming our curriculum learning strategy (see Section 3.4.1 for details).

For Stage 2, the objective is to ensure a representative training sample for the ranker. Since *SKILLSKAPE* consists of artificially generated data, we decided to incorporate both validation sets of *TECH* and *HOUSE* datasets into training. Given their relatively small size, we further expand the training data for Stage 2 by the *TECHWOLF* dataset. We acknowledge that such a decision prevents assessing the performance of the ranking stage on this dataset. However, such a step was crucial due to the unique nature of real-life data, where job description fragments can consist of both single phrases (e.g., "Python") as well as longer texts describing one or multiple competencies. Section 3.5 describes the exact process of forming training data for this stage.

**Table 1**
Characteristics of training and testing data used in our method. *Synth.* refers to artificially created datasets whereas *Tax.* indicates Taxonomy origin. *S1* and *S2* describe Dense Skill Retriever (bi-encoder) and Binary Skill Ranker (cross-encoder) stages, respectively. The ✓ and ✗ specify at what stage data was used.

| Dataset | Type | Size | S1 Train | S1 Test | S2 Train | S2 Test |
|---|---|---|---|---|---|---|
| *DECORTE* | Synth. | ~138K | ✓ (1 per skill) | ✗ | ✗ | ✗ |
| *SKILLSKAPE* (train) | Synth. | 6352 | ✓ | ✗ | ✓ | ✗ |
| *SKILLSKAPE* (val) | Synth. | 1316 | ✗ | ✗ | ✓ | ✗ |
| *SKILLSKAPE* (test) | Synth. | 1272 | ✗ | ✓ | ✗ | ✓ |
| *ESCO-D* | Tax. | ~112k | ✓ | ✗ | ✗ | ✗ |
| *HOUSE* (val) | Real | 131 | ✗ | ✗ | ✓ | ✗ |
| *HOUSE* (test) | Real | 532 | ✗ | ✓ | ✗ | ✓ |
| *TECH* (val) | Real | 152 | ✗ | ✗ | ✓ | ✗ |
| *TECH* (test) | Real | 644 | ✗ | ✓ | ✗ | ✓ |
| *TECHWOLF* | Real | 588 | ✗ | ✓ | ✓ | ✗ |

## 3.4. Stage 1: Bi-encoder for Skill Retrieval

Our bi-encoder is based on all-mpnet-base-v2[1], a sentence transformer model pre-trained for semantic similarity tasks. The model has previously demonstrated its effectiveness in the job domain for job recommendation [31] and skill extraction [2] problems.

[1]https://huggingface.co/sentence-transformers/all-mpnet-base-v2

The bi-encoder processes inputs independently, encoding job description sentences and skill labels into a shared embedding space. During inference, we pre-compute embeddings for all 13,890 ESCO skills, enabling real-time retrieval via similarity search. For each query sentence, the model retrieves the top-K skills based on cosine similarity scores.

The data is organised into pairs of job description fragments and their assigned skill labels. For multi-label sentences, we create one query–skill pair per gold label. We employ the augmentation strategy from [2], where sentences are randomly concatenated during training. To prevent augmented sentence pairs from serving as negatives to each other, we maintain a mask that excludes these pairs (and associated skill labels) from the negative set. As examined in [2], this augmentation strategy forces the model to learn robust representations. The model must identify relevant skills even when unrelated content is present. This mirrors real job descriptions, where target skills are often embedded among other skills and irrelevant text. The augmentation strategy is only applied to job description sentences and not skill labels at this stage.

We employ NT-Xent loss from [32] as a learning objective. Following the approach used in CLIP [33], we compute the loss symmetrically. The loss for a single direction is defined as:

$$L_{q\to s} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(s_{i,p_i}/\tau)}{\sum_{j=1}^{N} M_{i,j}\exp(s_{i,j}/\tau)}, \tag{1}$$

where $N$ is the batch size, $s_{i,j} = \cos(q_i, s_j)$ represents the cosine similarity between the $i$-th query embedding $q_i$ and the $j$-th skill embedding $s_j$, and $p_i$ denotes the index of the positive skill for query $i$. The mask $M_{i,j} = 0$ when skill $j$ should be excluded (i.e., $j = p_i$ or $j$ comes from an augmented version of query $i$), and $M_{i,j} = 1$ otherwise. The temperature parameter $\tau$ controls the sharpness of the distribution. Total loss is:

$$L = \frac{1}{2}(L_{q\to s} + L_{s\to q}), \tag{2}$$

where $L_{s\to q}$ is computed identically as in (1) but with skill labels as anchors and queries as positives/negatives. This bidirectional formulation ensures that both job descriptions and skills are equally optimised within the shared embedding space.

**Experimental Configuration.** The AdamW optimiser is used with a cosine learning rate schedule and a base learning rate of 2e-5, with 5% of training steps for warmup. We adopt cosine decay as it provided smoother convergence than a linear schedule in preliminary runs. The model is trained for a single epoch, with a batch size of 64 (the largest fitting on a 16GB GPU), and gradients clipped at 1.0 to stabilise training against large updates. Training is performed in mixed precision to improve efficiency. The temperature parameter is set to 0.05, selected via grid search on the *SKILLSKAPE* validation set in increments of 0.01 within [0.01, 0.07]. For tokenization, we set the maximum token length to 128 and 32 for sentences and skill labels, respectively. With the average example length in *SKILLSKAPE* validation set of 27.8 words and ESCO skill labels no longer than 13 words, this ensures complete context coverage while maintaining computational efficiency.

### 3.4.1. Curriculum Learning with Skill Definitions

Prior to the training procedure described above, we employ a pre-training phase that leverages definitions and

all available skill labels from ESCO (*ESCO-D*). Together, these form our curriculum training strategy, where the model first learns from simpler skill-definition alignments before progressing to more complex job description-skill mappings. During pre-training, we train the bi-encoder on skill-definition pairs using the same symmetric NT-Xent loss and augmentation strategy (applied to definitions) as in the main phase. This ensures consistency between pre-training and fine-tuning phases while teaching the model to align skill names with their semantic meanings. Notably, we reuse ESCO definitions in the main training phase, where they serve as high-quality reference examples alongside job descriptions, contributing to improved performance as shown in our ablation studies (see Section 5).

**Experimental Configuration.** Apart from temperature and learning rate, all of the hyperparameters from the main training phase are retained. We apply a higher learning rate of 3e-5 to accelerate learning of skill semantics during the pre-training phase. The temperature is set to 0.03, determined through validation experiments similar to the main training phase.

### 3.5. Stage 2: Cross-encoder for Skill Ranking

At the base of the cross-encoder, we adopt the ms-marco-MiniLM-L6-v2 model[2]. It was tuned for the ranking and displays a strong efficiency–effectiveness trade-off via self-attention distillation [34].

The training data used for this stage is directly sourced from the previous dense retrieval stage. Specifically, for each job description sentence in training data, we retrieve the top-100 skill candidates. To ensure positives are present, we inject any missing gold skills by replacing the lowest-scoring retrieved items. This is unique to the training data, as at inference, the test sets contain only the originally retrieved skills. Such a configuration represents a more realistic setting where a dedicated ranker might not have access to a complete set of true labels.

For each sentence, we pair it with all candidate skills and assign a binary label (1 if the skill appears in the gold set, 0 otherwise). To improve generalisation, we apply two lightweight augmentations with probability 0.2: (i) partial label masking, where one token of a multi-word skill is replaced by a [MASK] placeholder to discourage memorisation of exact surface forms; and (ii) sentence word dropout, where one random token is removed from longer sentences to add noise.

During training, each job description sentence is paired with 100 candidate skills, of which at most 10 are relevant, yielding a highly imbalanced label distribution. To mitigate the dominance of easy negatives, we replace standard binary cross-entropy with the focal loss [35]. Focal loss down-weights well-classified examples, forcing the model to focus on hard positives and hard negatives. Let $z_i$ be the logit and $y_i \in \{0, 1\}$ the label. The focal loss is:

$$\mathscr{L}_{\text{focal}} = \frac{1}{N} \sum_{i=1}^{N} \alpha_t (1 - p_t)^{\gamma} [-\log p_t],$$

with

$$p_t = \begin{cases} \sigma(z_i) & \text{if } y_i = 1, \\ 1 - \sigma(z_i) & \text{if } y_i = 0, \end{cases} \qquad \alpha_t = \begin{cases} \alpha & \text{if } y_i = 1, \\ 1 - \alpha & \text{if } y_i = 0, \end{cases}$$

[2]https://huggingface.co/cross-encoder/ms-marco-MiniLM-L6-v2

where $\gamma$ controls the degree of focussing and $\alpha$ balances positive vs. negative classes. To further address class imbalance during training, we employ a balanced batch sampler that maintains approximately 30% positive examples per batch. This prevents the model from simply learning to predict all negatives.

**Experimental Configuration.** We split the constructed dataset 80:20 into training and validation and train the model for 5 epochs with AdamW (learning rate 2e-5) and a linear warm-up of 10% of total steps. Validation tests showed no consistent benefits beyond 5 epochs. Gradients are clipped at 1.0 with batch size set to 64.

At inference, the model outputs a relevance score per sentence-skill pair for each test set. While training uses top-100 candidates, we evaluate on top-20 across all methods for practical reasons: managing API costs for LLM baselines and maintaining reasonable inference speed. The decision threshold is selected on a held-out split of the constructed Stage 2 training pool, tested in increments of 0.05 in the range [0.1, 0.7]. Based on this tuning, a fixed threshold of 0.2 is used for all reported results. Similarly, we fix $\alpha$ and $\gamma$ at 0.8 and 3.0, respectively, testing multiple configurations [0.5, 0.6, 0.8, 0.9] ($\alpha$) and [2.0, 2.5, 3.0] ($\gamma$). The maximum tokenization length is set to 128 for each sentence-skill pair.

## 4. Results and Discussion

While our pipeline can operate as an integrated system, we evaluate the bi-encoder and cross-encoder components separately on the datasets described in Section 3.3. This is done to understand their individual contributions and identify potential bottlenecks.

### 4.1. Evaluation Metrics and Benchmarks

We evaluate each pipeline component with task-appropriate metrics and baselines.

**Bi-encoder Evaluation.** Following established practice in skill extraction tasks [19, 2, 8], we employ R-Precision@K (RP@K) and Mean Reciprocal Rank (MRR). Since job description sentences typically contain at most 10 relevant skills, we report RP@5 and RP@10. For $N$ job description sentences, where $R_n$ is the number of gold ESCO skills for sentence n, and $\text{Rel}(n, k) \in \{0, 1\}$ indicates whether the k-th predicted skill is relevant (binary indicator), RP@K is defined as:

$$\text{RP@}K \;=\; \frac{1}{N} \sum_{n=1}^{N} \frac{1}{\min(K,\, R_n)} \sum_{k=1}^{K} \text{Rel}(n, k).$$

As baselines, we compare against: (1) the base all-mpnet-base-v2 model without fine-tuning (BASE), and (2) a similar skill extraction method from Decorte et al. [2].

**Cross-encoder Evaluation.** Our cross-encoder performs binary classification on retrieved candidates, predicting whether each ESCO skill is relevant to the given job description sentence. We evaluate using the micro-F1 score as it captures both the identification of relevant skills and the rejection of irrelevant ones [3]. For baselines, we implement LLM-based ranking using GPT-4o-mini and GPT-4.1, building on similar approaches [8]. Each model receives

**Table 2**
Retrieval performance (RP@5 / RP@10 / MRR, %) of the bi-encoder retriever. BASE = all-mpnet-base-v2 (no fine-tuning). Numbers for Decorte et al. [2] are taken from the original paper; "–" denotes not reported. Our results are mean ± standard deviation over 3 random seeds. Best results per dataset are highlighted in **bold**.

| | *HOUSE* | | | *TECH* | | | *TECHWOLF* | | | *SKILLSKAPE* | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RP@5 | RP@10 | MRR | RP@5 | RP@10 | MRR | RP@5 | RP@10 | MRR | RP@5 | RP@10 | MRR |
| BASE (all-mpnet-base-v2) | 26.17 | 36.76 | 26.19 | 39.3 | 49.97 | 38.69 | 33.17 | 41.82 | 29.22 | 29.39 | 39.5 | 36.47 |
| Decorte et al. [2] | 45.74 | - | 42.75 | 54.62 | - | 52.85 | 54.57 | - | 52.55 | - | - | - |
| Ours (curriculum bi-encoder) | **49.14± 0.46** | **59.61± 1.34** | **48.39± 0.59** | **59.40± 1.30** | **69.93± 1.35** | **57.26± 0.39** | **56.52± 0.84** | **64.91± 0.65** | **53.61± 0.13** | **62.02± 0.21** | **73.15± 0.11** | **72.46± 0.51** |

the query and top-20 retrieved candidates with instructions to classify each as relevant or irrelevant using single-shot prompting with temperature set to 0 for deterministic outputs. The demonstrations for LLM baselines are drawn from training data, selected to maximise overlap with the candidate set (see Appendix A for exact configuration).

## 4.2. Bi-encoder Performance

Table 2 presents our bi-encoder results across four skill extraction datasets. Our curriculum-based approach consistently outperforms both baselines, achieving the highest scores on all metrics.

Compared to Decorte et al. [2], we observe improvements ranging from 1.95 percentage points (pp) (*TECHWOLF*) to 4.78 pp (*TECH*) in RP@5. The gains are even more substantial against the non-fine-tuned baseline, with *SKILLSKAPE* experiencing a 32.63 pp improvement in RP@5. The consistent gains from RP@5 to RP@10 indicate that additional relevant skills are retrieved when expanding the candidate set. However, the gap between MRR scores (48.39-72.46%) and perfect ranking indicates that while most relevant skills are retrieved, they are not always optimally ordered. This motivates our cross-encoder stage, which can take advantage of seeing more closely associated candidates in determining the relevance of the skills.

Our curriculum bi-encoder achieves consistent improvements across all datasets, showcasing the coverage of various job domains present in evaluation data. The small standard deviations (typically <1.5 pp) across three random seeds indicate stable training despite the additional complexity of our curriculum setup. The specific contribution of the pre-training phase is analysed in Section 5.

### 4.2.1. Zero-shot Performance

To evaluate our model's ability to handle emerging skills not present in training data, we conduct zero-shot experiments on held-out skills. We fix a held-out skill set $H$ of 100 ESCO skills and exclude $H$ from all training data (both pre-training and fine-tuning). Specifically, we exclude the 50 most frequent and 50 least frequent skills based on the *SKILLSKAPE* test set. This selection ensures we test on both common skills (that the model might implicitly learn through co-occurrences) and rarer ones. At inference, the retriever still searches the full ESCO taxonomy. For each test set we filter to queries $q$ whose gold labels intersect $H$, and treat only held-out labels as relevant.

Table 3 shows that our model successfully retrieves held-out skills despite no direct training exposure. Improvements over the non-fine-tuned all-mpnet-base-v2 range from 11.02 pp (*TECH*) to 25.88 pp (*HOUSE*) in RP@5. A similar pattern can be observed for MRR scores with our model providing up to 19.33 pp (*HOUSE*) improvement. These results demonstrate that our approach creates skill representations that generalise beyond the training vocabulary.

Notably, zero-shot performance shows higher variance across seeds (up to 3.29 pp standard deviation) compared to the complete model (<1.5 pp). This aligns with prior work showing increased instability in low-data regimes [36], where different initialisations lead to different representation geometries for unseen classes, especially with a low amount of training iterations.

**Table 3**
Zero-shot retrieval performance (RP@5 / RP@10 / MRR, %). BASE = all-mpnet-base-v2 (no fine-tuning). "OURS" is a zero-shot trained bi-encoder and reports mean ± standard deviation over 3 random seeds. Evaluation is restricted to queries associated with the 100 held-out ESCO skills (see text). Best results per dataset are highlighted in **bold**.

| | BASE | | | OURS | | |
|---|---|---|---|---|---|---|
| | RP@5 | RP@10 | MRR | RP@5 | RP@10 | MRR |
| *HOUSE* | 24.34 | 36.18 | 16.95 | **50.22± 2.49** | **62.94± 2.31** | **36.28± 1.09** |
| *TECH* | 42.91 | 51.88 | 34.64 | **53.93± 2.58** | **62.86± 2.17** | **41.14± 1.48** |
| *TECHWOLF* | 35.9 | 38.46 | 24.16 | **51.95± 3.29** | **62.39± 1.48** | **33.73± 1.45** |
| *SKILLSKAPE* | 18.12 | 29.27 | 17.31 | **32.54± 1.47** | **45.18± 0.72** | **30.02± 0.12** |

## 4.3. Cross-Encoder Performance

Building on our bi-encoder's strong retrieval performance, we now evaluate the cross-encoder stage that refines these candidates through binary relevance classification. As explained in Section 3.5, we train on top-100 retrieved candidates to ensure broad coverage and evaluate on top-20 for practical inference and fair comparison to our LLM baselines. The results are provided in Table 4.

Our fine-tuned cross-encoder delivers the best performance across all three selected benchmarks. These benefits are more pronounced when compared to a simpler GPT-4o-mini model, ranging from 6.39 pp (*HOUSE*) to 30.54 pp (*SKILLSKAPE*) increase in F1 scores. However, even when paired with a much more capable GPT-4.1 model, our ranker provides improvements for *TECH* (+8.03 pp) and *SKILLSKAPE* (+22.54 pp), with only marginal gains in *HOUSE* (+ 0.53 pp).

Beyond performance advantages, our cross-encoder offers significant practical benefits for large-scale deployment. In our setup, GPT-4o-mini costs ≈$0.0001/example and GPT-

**Table 4**
Performance of the cross-encoder ranker (micro-F1, %). All methods rank the same top-20 candidates from Stage 1. We report mean ± standard deviation over 3 random seeds. Best results per dataset are highlighted in **bold**.

| | *HOUSE* | *TECH* | *SKILL-SKAPE* |
|---|---|---|---|
| GPT-4o-mini | 26.60 ± 0.46 | 27.91 ± 0.45 | 35.11 ± 0.09 |
| GPT-4.1 | 32.46 ± 0.39 | 35.80 ± 0.15 | 43.11 ± 0.13 |
| Ours (cross-encoder) | **32.99 ± 1.18** | **43.83 ± 0.63** | **65.65 ± 0.58** |

4.1 ≈$0.001/example[3], while our cross-encoder has a one-time training cost and near-zero per-example inference cost. For labour-market pipelines processing millions of postings, this difference is material. Furthermore, our dedicated ranker achieves approximately 0.021s per example inference time, compared to 1.07s average for LLM-based solutions, a ≈50x speed improvement (see Appendix B for full breakdown of run-times). While open-source alternatives like Llama exist [37], models matching GPT-4's ranking performance require high-end GPUs (e.g., A100), whereas our ranker runs efficiently on RTX-4070Ti SUPER with 16GB of available memory.

Our results demonstrate the value of dedicated ranking with our current bi-encoder. Notably, concurrent work [21] has achieved even stronger retrieval performance, which presents an exciting opportunity: combining state-of-the-art retrieval with our cross-encoder could yield substantially better results. At inference (top-20 skill candidates), the retrieved list contains, on average, about 78%[4] of the gold skills per sentence across the Stage 2 evaluation sets. Therefore, roughly 22% of gold skills are absent from the ranker's candidate set. Improved retrieval would provide our ranker with more complete candidate sets, likely amplifying its performance.

## 5. Ablation Studies

Table 5 presents ablations on three key design choices: (1) number of *DECORTE* examples, (2) inclusion of *SKILLSKAPE*, and (3) use of *ESCO* definitions with vs. without pre-training phase.

Using more *DECORTE* examples provides no benefit (and sometimes hurts performance), validating our decision to use only one example per skill from this dataset. The augmentation strategy from [2] appears to provide sufficient diversity without needing multiple examples.

Furthermore, we observe that the addition of *SKILLSKAPE* not only offers improvements in its corresponding test set (+11.8 pp) but also boosts the performance in *HOUSE* (+3.59 pp) and *TECH* (+3.94 pp) when compared to the model using a single example from *DECORTE*. Even with limited taxonomy coverage, exposure to sentences describing more than one skill benefits the training.

Interestingly, our ablations reveal an interplay between ESCO definitions and the pre-training phase within the curriculum training regimen. Adding definitions to the training data without a pre-training phase provides inconsistent results, even decreasing performance in *TECH* by 1.22 pp. However, when pre-training is applied to this data with definitions, we observe consistent improvements across all datasets, most notably in *TECHWOLF* (+3.13 pp) and *TECH* (+2.22 pp). Comparing our full system to the model without definitions or pre-training, we achieve gains ranging from 1.02 pp (*SKILLSKAPE*) to 3.70 pp (*TECHWOLF*). This demonstrates that the introduced pre-training phase is essential for leveraging taxonomic knowledge, as without it, definitions can have a detrimental effect on retrieval performance. The modest but consistent gains justify using the full curriculum training and ESCO definitions as additional reference examples in our final architecture.

**Table 5**
Ablations on the bi-encoder retriever (RP@5, %). We report the mean ± standard deviation over 3 seeds and focus on RP@5 for brevity. Best results per dataset are highlighted in **bold**.

| | *HOUSE* | *TECH* | *TECH-WOLF* | *SKILL-SKAPE* |
|---|---|---|---|---|
| Ours (curriculum bi-encoder) | **49.14 ± 0.46** | **59.40 ± 1.30** | **56.52 ± 0.84** | **62.02 ± 0.21** |
| Ablation Configurations | | | | |
| *DECORTE* (1 example) | 44.70 ± 0.17 | 54.46 ± 0.46 | 52.83 ± 0.46 | 49.20 ± 0.53 |
| *DECORTE* (10 examples) | 42.87 ± 0.44 | 54.47 ± 0.32 | 52.57 ± 0.44 | 49.46 ± 0.62 |
| *DECORTE* (1 example) + *SKILLSKAPE* | 48.29 ± 0.39 | 58.40 ± 0.48 | 52.82 ± 0.57 | 61.00 ± 0.38 |
| *DECORTE* (1 example) + *SKILLSKAPE* + *Definitions* (no pre-training) | 48.29 ± 1.15 | 57.18 ± 0.76 | 53.39 ± 0.29 | 61.99 ± 0.30 |

## 6. Limitations and Future Work

Our two-stage architecture requires training separate models. This increases computational requirements compared to single-stage retrieval. While the cross-encoder processes queries slower than bi-encoder retrieval due to joint encoding, it remains efficient at 0.021s per example. Combined with its more concise outputs (specific skills vs a list of relevant candidates) and the fact that real-time skill extraction is rarely critical in labour market analysis, the architecture is well-suited for practical deployment.

Data scarcity remains a key challenge. Limited availability of high-quality annotated job descriptions necessitated using validation sets for cross-encoder training. While we maintained evaluation integrity by using separate test sets, larger training corpora would likely improve performance.

Our current evaluation is limited to English-language job postings. Generalisation to other languages and industries requires further investigation. ESCO's availability in 28 languages presents an opportunity for multilingual extension, following recent work in multilingual job recommendation [31].

Finally, our cross-encoder uses direct label encoding rather than one-hot representations. In theory, this enables handling of new skills, though cross-encoder generalisation capability to previously unseen skill taxonomies requires further empirical validation. Future work should explore cross-taxonomy transfer and the framework's ability to adapt to evolving skill landscapes.

[3]Based on June 2025 prices, averaged over all evaluation data and runs. Total replication costs for our full evaluation are $0.61 (GPT-4o-mini) $5.88 (GPT-4.1).
[4]Measured as Recall@20, macro-averaged.

## 7. Conclusions

This paper introduced the two-stage neural architecture for skill extraction, adapting successful information retrieval practices to address the unique challenges of matching job descriptions to large skill taxonomies. By combining bi-encoder retrieval with cross-encoder ranking, our approach bridges the gap between efficient candidate generation and precise skill identification.

Our experiments validate the effectiveness of the two-stage approach across multiple dimensions. The bi-encoder achieves up to 62.02% RP@5 through curriculum learning with ESCO definitions, while maintaining strong zero-shot capability compared to the vanilla model on held-out skills. More importantly, our cross-encoder ranker amplifies these gains, delivering F1 scores up to 30.54 percentage points higher than LLM-based alternatives. Notably, our ablations revealed that in most cases, taxonomic definitions provide value only through structured pre-training, highlighting the importance of curriculum design in leveraging existing resources. Together, these components create a system that balances practical deployability with strong skill extraction performance.

These results showcase how two-stage recommender architectures can effectively address skill extraction challenges in HR systems. While previous work has explored retrieval-based approaches or LLM-based ranking for skill extraction, ours is the first to train purpose-built neural architectures for both retrieval and ranking stages within a unified framework. This modular design can leverage advances in either component, with better retrievers yielding richer candidate sets, and stronger rankers delivering more precise relevance judgments. HR applications increasingly rely on recommender techniques across recruitment (job-candidate matching), development (skill gap identification), and retention (career path recommendations). Our two-stage approach offers a flexible foundation that can be integrated into these diverse recommendation pipelines, where extracted skills often serve as essential features for downstream tasks.

## Acknowledgments

We would like to thank Franziska Heck for her feedback on this paper. We also thank Katie Killen, Alistair Lawson, Dimitra Gkatzia and Matthew Dutton for their support in this project. This research was supported by the Economic and Social Research Council (Grant Ref: ES/P000681/1).

## Ethical Considerations

We evaluate on publicly available datasets. Likewise, our training data comprises job advertisements and taxonomy labels publicly available; we do not process personal data. This publication uses the ESCO classification of the European Commission. All other datasets are accessible through CC-BY-4.0 (*DECORTE*, *HOUSE*, *TECH*, *TECHWOLF*) and MIT licences (*SKILLSKAPE*). Nonetheless, job ads and standardised taxonomies may encode societal and market biases (e.g., occupational stereotypes). Consequently, our system is intended for labour-market analytics and skill insights rather than automated hiring decisions. We recommend human oversight for any downstream HR use. We release code and configuration files, to support reproducibility.

## Declaration on Generative AI

During the preparation of this work, the authors used *Grammarly*, *ChatGPT* in order to: *Grammar and spelling check*, *Paraphrase and reword.* After using these tools, the authors reviewed and edited the content and take full responsibility for the publication's content.

## References

[1] W. E. Forum, The Future of Jobs Report 2025, Technical Report, Amherst, MA, USA, 2025.

[2] Decorte, Jens-Joris and Verlinden, Severine and Van Hautte, Jeroen and Deleu, Johannes and Develder, Chris and Demeester, Thomas, Extreme multi-label skill extraction training using large language models, in: AI4HR PES, ECML-PKDD 2023 Workshop, Proceedings, 2023, pp. 1–10.

[3] A. Magron, A. Dai, M. Zhang, S. Montariol, A. Bosselut, JobSkape: A framework for generating synthetic job postings to enhance skill matching, in: E. Hruschka, T. Lake, N. Otani, T. Mitchell (Eds.), Proceedings of the First Workshop on Natural Language Processing for Human Resources (NLP4HR 2024), Association for Computational Linguistics, St. Julian's, Malta, 2024, pp. 43–58. URL: https://aclanthology.org/2024.nlp4hr-1.4/.

[4] A. Gugnani, H. Misra, Implicit skills extraction using document embedding and its use in job recommendation, in: Proceedings of the AAAI conference on artificial intelligence, volume 34, 2020, pp. 13286–13293.

[5] E. Senger, M. Zhang, R. van der Goot, B. Plank, Deep learning-based computational job market analysis: A survey on skill extraction and classification from job postings, in: E. Hruschka, T. Lake, N. Otani, T. Mitchell (Eds.), Proceedings of the First Workshop on Natural Language Processing for Human Resources (NLP4HR 2024), Association for Computational Linguistics, St. Julian's, Malta, 2024, pp. 1–15. URL: https://aclanthology.org/2024.nlp4hr-1.1/.

[6] Z. Xu, Y. Chen, B. Hu, M. Zhang, A read-and-select framework for zero-shot entity linking, in: H. Bouamor, J. Pino, K. Bali (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2023, Association for Computational Linguistics, Singapore, 2023, pp. 13657–13666. URL: https://aclanthology.org/2023.findings-emnlp.912/. doi:10.18653/v1/2023.findings-emnlp.912.

[7] J. Song, C. Jin, W. Zhao, A. McCallum, J.-Y. Lee, Comparing neighbors together makes it easy: Jointly comparing multiple candidates for efficient and effective retrieval, in: Y. Al-Onaizan, M. Bansal, Y.-N. Chen (Eds.), Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Miami, Florida, USA, 2024, pp. 22255–22269. URL: https://aclanthology.org/2024.emnlp-main.1242/. doi:10.18653/v1/2024.emnlp-main.1242.

[8] B. Clavié, G. Soulié, Large language models as batteries-included zero-shot esco skills matchers, in: Proceedings of the 3rd Workshop on Recommender Systems for Human Resources (RecSys in HR 2023), in conjunction with the 16th ACM Conference on Recommender Systems, Association for Computing Machinery, Singapore, 2023.

[9] K. D'Oosterlinck, O. Khattab, F. Remy, T. Demeester, C. Develder, C. Potts, In-context learning for extreme multi-label classification, arXiv preprint arXiv:2401.12178 (2024).

[10] M. Zhang, K. Jensen, S. Sonniks, B. Plank, SkillSpan: Hard and soft skill extraction from English job postings, in: M. Carpuat, M.-C. de Marneffe, I. V. Meza Ruiz (Eds.), Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Seattle, United States, 2022, pp. 4962–4984. URL: https://aclanthology.org/2022.naacl-main.366. doi:10.18653/v1/2022.naacl-main.366.

[11] M. Zhang, R. van der Goot, M.-Y. Kan, B. Plank, NNOSE: Nearest neighbor occupational skill extraction, in: Y. Graham, M. Purver (Eds.), Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, St. Julian's, Malta, 2024, pp. 589–608. URL: https://aclanthology.org/2024.eacl-long.35.

[12] K. Nguyen, M. Zhang, S. Montariol, A. Bosselut, Rethinking skill extraction in the job market domain using large language models, in: E. Hruschka, T. Lake, N. Otani, T. Mitchell (Eds.), Proceedings of the First Workshop on Natural Language Processing for Human Resources (NLP4HR 2024), Association for Computational Linguistics, St. Julian's, Malta, 2024, pp. 27–42. URL: https://aclanthology.org/2024.nlp4hr-1.3.

[13] A. Herandi, Y. Li, Z. Liu, X. Hu, X. Cai, Skill-llm: Repurposing general-purpose llms for skill extraction, arXiv preprint arXiv:2410.12052 (2024).

[14] I. Konstantinidis, M. Maragoudakis, I. Magnisalis, C. Berberidis, V. Peristeras, Knowledge-driven unsupervised skills extraction for graph-based talent matching, in: Proceedings of the 12th Hellenic Conference on Artificial Intelligence, 2022, pp. 1–7.

[15] N. Goyal, J. Kalra, C. Sharma, R. Mutharaju, N. Sachdeva, P. Kumaraguru, Jobxmlc: Extreme multi-label classification of job skills with graph neural networks, in: Findings of the Association for Computational Linguistics: EACL 2023, 2023, pp. 2181–2191.

[16] A.-s. Gnehm, E. Bühlmann, H. Buchs, S. Clematide, Fine-grained extraction and classification of skill requirements in German-speaking job ads, in: D. Bamman, D. Hovy, D. Jurgens, K. Keith, B. O'Connor, S. Volkova (Eds.), Proceedings of the Fifth Workshop on Natural Language Processing and Computational Social Science (NLP+CSS), Association for Computational Linguistics, Abu Dhabi, UAE, 2022, pp. 14–24. URL: https://aclanthology.org/2022.nlpcss-1.2/. doi:10.18653/v1/2022.nlpcss-1.2.

[17] D. C. Kavargyris, K. Georgiou, E. Papaioannou, K. Petrakis, N. Mittas, L. Angelis, Escox: A tool for skill and occupation extraction using llms from unstructured text, Software Impacts (2025) 100772.

[18] M. Zhang, K. N. Jensen, R. van der Goot, B. Plank, Skill extraction from job postings using weak supervision, in: Proceedings of RecSys in HR'22: The 2nd Workshop on Recommender Systems for Human Resources, in conjunction with the 16th ACM Conference on Recommender Systems, Association for Computing Machinery, Seattle, 2022.

[19] J.-J. Decorte, J. Van Hautte, J. Deleu, C. Develder, T. Demeester, Design of negative sampling strategies for distantly supervised skill extraction, in: Kaya, Mesut and Bogers, Toine and Graus, David and Mesbah, Sepideh and Johnson, Chris and Gutiérrez, Francisco (Ed.), Proceedings of the 2nd Workshop on Recommender Systems for Human Resources (RecSys-in-HR 2022), volume 3218, CEUR, 2022, p. 7. URL: https://ceur-ws.org/Vol-3218/RecSysHR2022-paper_4.pdf.

[20] M. Zhang, R. van der Goot, B. Plank, ESCOXLM-R: Multilingual taxonomy-driven pre-training for the job market domain, in: A. Rogers, J. Boyd-Graber, N. Okazaki (Eds.), Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Toronto, Canada, 2023, pp. 11871–11890. URL: https://aclanthology.org/2023.acl-long.662/. doi:10.18653/v1/2023.acl-long.662.

[21] J.-J. Decorte, J. Van Hautte, C. Develder, T. Demeester, Efficient text encoders for labor market analysis, arXiv preprint arXiv:2505.24640 (2025).

[22] P. Soviany, R. T. Ionescu, P. Rota, N. Sebe, Curriculum learning: A survey, International Journal of Computer Vision 130 (2022) 1526–1565.

[23] Z. Ma, Z. Dou, W. Xu, X. Zhang, H. Jiang, Z. Cao, J.-R. Wen, Pre-training for ad-hoc retrieval: Hyperlink is also you need, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, CIKM '21, Association for Computing Machinery, New York, NY, USA, 2021, p. 1212–1221. URL: https://doi.org/10.1145/3459637.3482286. doi:10.1145/3459637.3482286.

[24] W. X. Zhao, J. Liu, R. Ren, J.-R. Wen, Dense text retrieval based on pretrained language models: A survey, ACM Transactions on Information Systems 42 (2024) 1–60.

[25] R. Nogueira, K. Cho, Passage re-ranking with bert, arXiv preprint arXiv:1901.04085 (2019).

[26] H. Dong, J. Chen, Y. He, Y. Liu, I. Horrocks, Reveal the unknown: Out-of-knowledge-base mention discovery with entity linking, in: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 452–462. URL: https://doi.org/10.1145/3583780.3615036. doi:10.1145/3583780.3615036.

[27] L. Zhang, D. Braun, Twente-BMS-NLP at PerspectiveArg 2024: Combining bi-encoder and cross-encoder for argument retrieval, in: Y. Ajjour, R. Bar-Haim, R. El Baff, Z. Liu, G. Skitalinskaya (Eds.), Proceedings of the 11th Workshop on Argument Mining (ArgMining 2024), Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 164–168. URL: https://aclanthology.org/2024.argmining-1.17/. doi:10.18653/v1/2024.argmining-1.17.

[28] W. Sun, L. Yan, X. Ma, S. Wang, P. Ren, Z. Chen, D. Yin, Z. Ren, Is ChatGPT good at search? investigating large language models as re-ranking agents, in: H. Bouamor, J. Pino, K. Bali (Eds.), Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, 2023, pp. 14918–14937. URL: https://aclanthology.org/2023.emnlp-main.923/. doi:10.18653/v1/2023.emnlp-main.923.

[29] S. Verma, F. Jiang, X. Xue, Beyond retrieval: Ensem-

bling cross-encoders and gpt rerankers with llms for biomedical qa, arXiv preprint arXiv:2507.05577 (2025).
[30] European Commission and Directorate-General for Employment, Social Affairs and Inclusion, ESCO handbook – European skills, competences, qualifications and occupations, Publications Office, 2017. doi:10.2767/934956.
[31] D. Deniz, F. Retyk, L. García-Sardiña, H. Fabregat, L. Gasco, R. Zbib, Combined unsupervised and contrastive learning for multilingual job recommendation, in: Proceedings of the 4th Workshop on Recommender Systems for Human Resources (RecSys-in-HR 2024), 2024.
[32] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: International conference on machine learning, PmLR, 2020, pp. 1597–1607.
[33] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., Learning transferable visual models from natural language supervision, in: International conference on machine learning, PmLR, 2021, pp. 8748–8763.
[34] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, M. Zhou, Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, Advances in neural information processing systems 33 (2020) 5776–5788.
[35] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2980–2988.
[36] M. Mosbach, M. Andriushchenko, D. Klakow, On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines, arXiv preprint arXiv:2006.04884 (2020).
[37] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al., Llama: Open and efficient foundation language models, arXiv preprint arXiv:2302.13971 (2023).

# A. Prompt Configuration for LLM-based ranker baselines

The created LLM-based rankers are deployed using the OpenAI API platform[5], implemented in Python. In total, two models: GPT-4o-mini and GPT-4.1 are tested. Each model is provided with the following system prompt:

> **System Prompt:** `You are an expert skill classifier. Given a sentence and a list of possible skills, your task is to select only the skills that are explicitly or implicitly required. Be precise and avoid including unrelated or weakly related skills. Return a JSON { "relevant_skills": ["skill_1", "skill_2", ...] }. If no skills are relevant, return { "relevant_skills": [] }. Do not add any other keys or text.`

This prompt defines the task and ensures that the output is constrained to a dictionary format, enabling efficient parsing and evaluation.

[5] https://openai.com/api/

At inference time, we provide a single-shot demonstration example using a designated `get_demonstration` function. This improves model performance and ensures fair comparability with the cross-encoder ranker, which also utilises training data. The function selects a demonstration example from a pool of annotated instances based on maximal skill overlap with the input (see (A)). The candidate lists used for demonstration are restricted to the top-20 labels prior to the injection of missing gold labels. This is done to control API cost and retain consistency with test examples.

**Algorithm 1** Get Demonstration Based on Skill Overlap

**Require:** List of skills $S$, reference data $D$
**Ensure:** A selected demonstration example

**if** $D$ is empty **then**
  **return** None
**end if**
$R \leftarrow$ set of skills $S$
$B \leftarrow$ empty list {Best examples}
$M \leftarrow -1$ {Max overlap}
**for all** $e \in D$ **do**
  $O \leftarrow |R \cap \text{set}(e.\text{candidate\_labels})|$
  **if** $O > M$ **then**
    $B \leftarrow [e]$
    $M \leftarrow O$
  **else if** $O = M$ **then**
    append $e$ to $B$
  **end if**
**end for**
**if** $M = 0$ **then**
  **return** random choice from $D$
**else**
  **return** random choice from $B$
**end if**

Given the original job description, a set of candidate skills, and the retrieved demonstration, the model predicts a list of truly relevant skills. These are then compared against gold labels using the same evaluation protocol as the cross-encoder ranker.

# B. Runtime Statistics

All experiments were run on a single RTX 4070Ti SUPER GPU with 16GB VRAM. Table 6 reports training time (total) and average per-example inference time. Times are averaged over 3 random seeds.

**Table 6**
Runtimes for the proposed two-stage pipeline. Training times are wall-clock totals; inference times are per-example averages over all evaluation sets at top-20 candidates. Stage 1 also incurs a one-time cost to encode all ESCO labels (≈2s).

| | Train | Test (inference) |
|---|---|---|
| Stage 1 | | |
| Bi-encoder (pre-train phase) | 8.6 minutes | <0.002 seconds per example |
| Bi-encoder (fine-tuning phase) | 7 minutes | <0.002 seconds per example |
| Stage 2 | | |
| Cross-encoder | 55 minutes | 0.021 seconds per example |
| LLM (GPT-4o-mini) | - | 1.11 seconds per example |
| LLM (GPT-4.1) | - | 1.07 seconds per example |