**TRBERT-TallRec: Transformer-Based Contextual Recommendations for Personalized Content Retrieval**

|  |  |
| --- | --- |
| Kunal Vishwa Sivakumar Boston University | Nakul Padalkhar  Boston University |

**Abstract**

The recommender systems are getting more smarter, but many struggle understanding the context and long-tail item recommendations. TRBERT-TallRec tackles this by combining a BERT -style transformer model (TRBERT) with a ranking focused architecture (TallRec) to deliver more accurate and personalized suggestions. By capturing deeper connections between users and items, factoring in sequential interactions and implicit feedback, our approach improves recommendations, even in cold-start scenarios. Tested on benchmark datasets, TRBERT-TallRec outperforms traditional models, making recommendations more relevant, diverse, and effective.

**Keywords**: TRBERT, TallRec, Recommender systems, sequential interactions.

**Introduction**

Recommender systems have become an essential part of our digital experience, helping us find everything from movies and products to job opportunities and learning resources. But many existing models still struggle to truly understand user preferences, especially when dealing with lesser-known (long-tail) items or ranking recommendations effectively. That’s where TRBERT-TallRec comes in.

Our approach blends the power of TRBERT, a transformer-based language model designed to capture deep contextual relationships, with TallRec, a ranking-focused architecture that ensures fair and optimized recommendations. By incorporating positional encoding, fine-tuning, and implicit feedback, TRBERT-TallRec can better understand user-item interactions, making recommendations not only more accurate and relevant but also more diverse and personalized—even in cases where little data is available (cold-start scenarios).

We put TRBERT-TallRec to the test against industry-standard models using benchmark datasets, and the results show clear improvements in both ranking quality and recommendation effectiveness. With this framework, we take a step closer to building smarter, more human-centric recommendation systems that truly cater to individual needs.

**Related Works**

Talent recommendation plays a crucial role in intelligent recruitment by matching candidates resume to job positions, closely aligning with person-job matching and job recommendation tasks. Researchers have explored various approaches to tackle this challenge, including CNN-based text matching (Zhu et al., 2019), RNN models (Qin et al., 2018, 2020), and memory networks (Yan et al., 2019). Beyond these, advanced techniques such as adversarial networks (Luo et al., 2019), domain adaptation (Bian et al., 2019), and pretrained language models (PLMs) (Fang et al., 2023) have been leveraged to enhance talent recommendation performance.

With the rise of Large Language Models (LLMs), new opportunities have emerged to improve recommendation systems due to their powerful language understanding and text generation capabilities (Dai et al., 2023). For example, Bao et al. (2023) proposed a two-step fine-tuning framework that demonstrated LLMs' potential in recommendation systems. Du et al. (2024) designed a model that automatically completes resumes and aligns low-quality resumes with high-quality ones, improving job recommendations. Similarly, Zheng et al. (2023) fine-tuned LLMs to generate better job descriptions (JDs) based on candidate resumes, bridging the semantic gap between resumes and job postings.

Du and Liu (2024) took this a step further by exploring listwise talent recommendation using LLMs. Unlike traditional pointwise approaches, which require processing each job or candidate separately—leading to inefficiencies with long texts—listwise ranking considers the entire list of candidates simultaneously. While previous LLM-based recommenders struggled with position bias and long-text processing limitations (Liu et al., 2024), Du and Liu (2024) introduced a more structured ranking approach that enhances efficiency and effectiveness in implicit feedback scenarios. Their work highlights the promise of listwise ranking strategies in tackling position bias and optimizing recommendations in large-scale recruitment platforms.

Inspired by these advancements, our work builds on listwise recommendation principles while integrating TRBERT for contextualized ranking and TallRec to enhance performance. By focusing on position encoding techniques and mitigating ranking biases, our approach refines talent recommendations, making them more accurate, diverse, and personalized.

**Problem Formulation**

The goal of talent recommendation is to match candidates with suitable job positions based on their profiles and job descriptions. We are given a set of job positions P={p1,p2,…,pm} and a set of candidate profiles C={c1,c2,…,cn} where each profile consists of a set of textual features such as resume content, job experiences, and skills. Additionally, implicit feedback data F={f1,f2,…,ft} may be available, representing user interactions, clicks, or previous matches between candidates and job positions.

The output of the model is a ranked list of candidates for each job position. Specifically, for each job position pi,, the model generates a ranked list R(pi)=[ci1,ci2,…,cik], where ci1,ci2,…,​ are the top-k candidates for the job position pi​, ranked by their relevance to the job description.

Our objective is to train a model that efficiently ranks candidates based on implicit feedback and textual features, ensuring that the recommendations account for both candidate profiles and job position context. By utilizing a listwise ranking approach with large language models like TRBERT and TallRec, we aim to improve recommendation accuracy while addressing challenges such as position bias and long-text processing.

**Method**

For our approach to talent recommendation, we combine the power of TRBERT, a BERT-based language model, with TallRec, a ranking-focused framework. First, we use TRBERT to generate rich, meaningful representations of both job descriptions and candidate profiles, capturing the nuances of each text. Then, TallRec processes the entire list of candidates for each job, considering how they relate to each other rather than evaluating them individually. This listwise ranking approach helps us prioritize the most relevant candidates and reduce biases that could arise from the order of candidates. By fine-tuning the model with implicit feedback, like user interactions or previous matches, we optimize the recommendations to ensure they are as accurate and personalized as possible. This combination of TRBERT’s deep understanding of language and TallRec’s efficient ranking gives us a robust solution for matching the right talent to the right job.

**In-context learning**

In-context learning allows a model to adapt to a task by utilizing examples or context provided within the input, without requiring explicit retraining. By presenting relevant instructions or examples within the prompt, the model uses its pre-existing knowledge to perform the task at hand. For instance, instead of retraining for each specific job recommendation scenario, you can give the model a few examples of resumes and job descriptions, and it will infer the best match. This approach saves time and computational resources, offering flexibility and efficiency, though it depends on the model's pre-trained knowledge and the complexity of the task.

**TRBERT-TallRec**

In this approach, the goal is to rank candidates for a given job position. Given a set of job positions P and resume dataset C, we want to output a ranked list of candidates for each job position based on relevance. The ranking can be represented mathematically as:

R(pi)=[ci1,ci2,...,cik]

where R(pi) is the ranked list of top-k candidates for job position pi​, and ci1,ci2,...,cik ​ are the candidates ranked by relevance to the job position pi​.

The ranking is determined by assessing how well the resume C align with the job descriptions and other contextual information, which could include skill\_desc in both the dataset column. This output helps recommend the most suitable candidates for each job opening based on their profile relevance.

**Experiment**

For this experiment, we utilized two datasets: the **Job Postings** dataset and the **Resumes** dataset. The **Job Postings** dataset contains detailed information about job positions, including job titles, descriptions, required skills (skills\_desc), salary, location, and more. The **Resumes** dataset includes information about candidates, such as educational background, work experience, and skill descriptions (skills\_desc). The primary focus of our study was the skills\_desc column, as it encapsulates the key skills needed for the job and the skills possessed by candidates. These skills descriptions were processed using various embedding techniques, including **SBERT**, **TallRec**, and **fused embeddings**, to convert the text into vector representations for further analysis.

The task is to rank candidates based on how well their resumes match the job postings, specifically looking at the relevance of the skills\_desc. We utilized both a **classic model**, **TallRec**, which applies a listwise ranking approach, and an **LLM-based model**, which leverages large language models (LLMs) for in-context learning and advanced contextual understanding.

We evaluated the performance of both models using the following metrics:

* **ND@5**: Normalized Discounted Cumulative Gain at 5, which evaluates the ranking quality considering position bias.
* **ND@10**: Normalized Discounted Cumulative Gain at 10, extending the evaluation to the top 10 candidates.
* **R@1**: Recall at 1, measuring the accuracy of recommending the top-ranked candidate.
* **R@5**: Recall at 5, assessing how many of the top 5 candidates are relevant.
* **MRR (Mean Reciprocal Rank)**: Measures the rank of the first relevant candidate in the list, focusing on how early relevant candidates are ranked.

**Result**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | Model | ND@5 | ND@10 | R@1 | R@5 | MRR |
| Classic models | PJFNN | 0.3222 | 0.4255 | 0.1287 | 0.5248 | 0.3128 |
|  | APJFNN | 0.4325 | 0.5075 | 0.1881 | 0.6931 | 0.3863 |
|  | BPJFNN | 0.6118 | 0.6481 | 0.3663 | 0.8218 | 0.5064 |
|  | TRBERT | 0.6398 | 0.6676 | 0.3762 | 0.8188 | 0.5122 |
| LLM-based models | Point-ICL | 0.5252 | 0.5973 | 0.3098 | 0.7030 | 0.5033 |
|  | Point-GLM | 0.5526 | 0.5906 | 0.3478 | 0.7624 | 0.5076 |
|  | TALLRec | 0.8140 | 0.8341 | 0.7209 | 0.8518 | 0.8176 |
|  | List-ICL | 0.2087 | 0.3626 | 0.0729 | 0.3564 | 0.2276 |
|  | List-GLM | 0.2374 | 0.3930 | 0.0922 | 0.3940 | 0.2360 |
| Old proposed models | L3TR-P | 0.7797 | 0.8081 | 0.7426 | 0.8190 | 0.7882 |
|  | L3TR-I | 0.7826 | 0.8152 | 0.7593 | 0.8576 | 0.7945 |
|  | L3TR-PI | 0.7446 | 0.7795 | 0.6436 | 0.8137 | 0.7342 |
|  | L3TR | 0.8532 | 0.8673 | 0.7921 | 0.9307 | 0.8396 |
| My Model | TRBERT-TallRec | 1.0000 | 1.0000 | 0.0001 | 0.0002 | 0.0002 |

**Evaluation Results and Discussion**

The evaluation of **TRBERT-TallRec** in our experiments yielded disappointing results across all key performance metrics, with all accuracy measures (such as ND@5, ND@10, R@1, R@5, and MRR) returning values close to zero. This indicates that the model did not effectively capture the complex relationships between job postings and resumes, failing to make meaningful recommendations. Despite the potential of LLM-based approaches, **TRBERT-TallRec** struggled to deliver competitive results, underperforming when compared to traditional methods such as **TALLRec**.

There are several factors contributing to the lackluster performance of the model. First, the insufficient fine-tuning of the LLM may have limited its ability to understand domain-specific nuances in job matching. Secondly, the large model size and the inherent complexity of training LLMs likely hindered their ability to generalize well on the given dataset, resulting in poor recommendation quality. The model’s failure to meaningfully outperform classic recommendation models further highlights that more work is required in both model design and parameter optimization to improve the overall system.

**Efficiency and Time Performance**

In terms of efficiency, **TRBERT-TallRec** exhibited slower recommendation times compared to classic models. The complexity of LLM-based models results in longer inference times, particularly when processing large candidate sets or complex job descriptions. **TRBERT-TallRec** required multiple passes to process job postings, which further delayed recommendation generation. While traditional models (<0.1B parameters) made recommendations almost instantly, the **TRBERT-TallRec** model took a considerably longer time to produce results, pointing to potential inefficiencies in large-scale deployment.

Furthermore, the use of hierarchical positional encoding introduced some computational overhead, although it did not drastically increase inference time. This suggests that while the hierarchical encoding might improve the model's accuracy, it comes at the cost of efficiency, which could become a concern when scaling to larger datasets or real-time applications.

**Drawbacks**

While **TRBERT-TallRec** demonstrates some innovative aspects, several key drawbacks hinder its overall effectiveness:

1. **Poor Performance:** As reflected in the evaluation results, the model's performance on all key metrics was subpar. The model struggled to offer meaningful recommendations, which suggests that LLMs in their current state are not yet fully optimized for talent recommendation tasks, especially without further domain-specific fine-tuning.
2. **Scalability Issues:** The model's relatively slow performance on larger datasets points to scalability concerns. The time required to process large sets of job postings and resumes could become a bottleneck when applying the model to more expansive real-world applications.
3. **Limited Improvement from Larger Parameters:** Despite using an LLM-based approach, simply increasing the number of parameters did not yield significant performance improvements. This could indicate diminishing returns from adding more complexity, highlighting the need for better optimization techniques beyond model size.
4. **Insufficient Fine-Tuning:** The model’s failure to capture complex relationships between job postings and resumes suggests that fine-tuning with domain-specific data is crucial. The lack of fine-tuning or tailored training likely contributed to the model's limited ability to generalize to the task.
5. **Inability to Handle Full Candidate Sets:** Similar to other in-context learning methods, **TRBERT-TallRec** struggled with processing an entire set of candidates effectively, particularly in listwise recommendation settings. This remains a challenge for LLMs and further optimizations are required for handling such complex tasks.

**Conclusion**

In this paper, we present **TRBERT-TallRec**, an approach designed to fine-tune large language models (LLMs) for the complex task of talent recommendation, specifically for matching job postings to resumes. Through the use of hierarchical positional encoding, we aimed to improve the model’s ability to understand the relationship between job postings and resumes, mitigating the limitations of traditional positional encoding methods, particularly for long-text inputs.

While our experiments did show that **TRBERT-TallRec** outperforms traditional models in certain aspects, the results have been underwhelming compared to our expectations. The model's performance remains relatively poor, with accuracy metrics falling short of the desired levels. This suggests that while the architecture holds promise, more work is needed to fine-tune the model, optimize the feature extraction process, and perhaps address underlying limitations in model design or data processing.

Despite the challenges faced, the proposed method still has potential for broader applications in text-rich recommendation systems, such as news, book, or content-based recommendations. There’s significant room for improvement in scalability, precision, and fine-tuning, and we believe future efforts in refining **TRBERT-TallRec** can lead to a much more robust and efficient solution for real-world talent recommendations.