You Only Forward Once: An Efficient Compositional Judging Paradigm

Tianlong Zhang[1,*], Hongwei Xue[2,†], Shilin Yan[2], Di Wu[2], Chen Xu[2], Yunyun Yang[1,‡]

[1]Harbin Institute of Technology, Shenzhen [2]Accio, Alibaba Group

24b958007@stu.hit.edu.cn

Abstract

*Multimodal large language models (MLLMs) show strong potential as judges. However, existing approaches face a fundamental trade-off: adapting MLLMs to output a single score misaligns with the generative nature of MLLMs and limits fine-grained requirement understanding, whereas autoregressively generating judging analyses is prohibitively slow in high-throughput settings. Observing that judgment reduces to verifying whether inputs satisfy a set of structured requirements, we propose **YOFO**, a template-conditioned method that judges all requirements in a single forward pass. Built on an autoregressive model, YOFO accepts a structured requirement template and, in one inference step, produces a binary yes/no decision for each requirement by reading the logits of the final token associated with that requirement. This design yields orders-of-magnitude speedups while preserving interpretability. Extensive experiments show that YOFO not only achieves state-of-the-art results on standard recommendation datasets, but also supports dependency-aware analysis—where subsequent judgments are conditioned on previous ones—and further benefits from post-hoc CoT.*

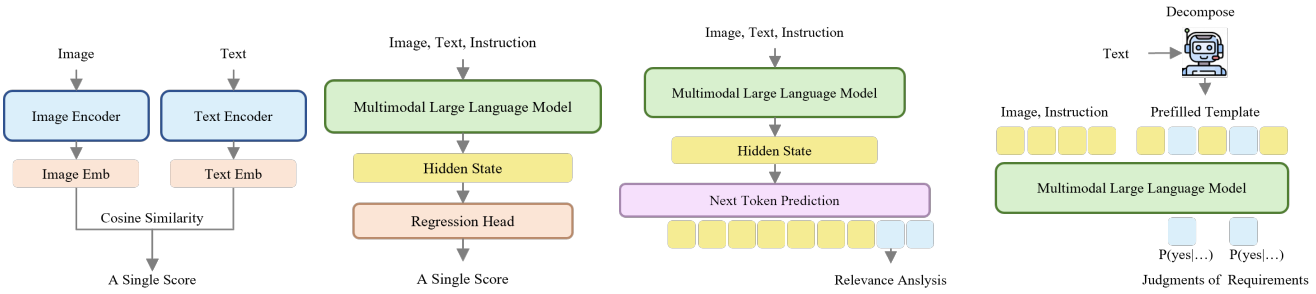
1. Introduction

Large language models (LLMs) [1, 3, 10, 19, 31, 38] have achieved remarkable progress, and advances in multimodal learning now enable them to process diverse modalities. Multimodal Large Language Models (MLLMs) [4, 5, 20, 21, 34] have recently demonstrated significant promise in information matching due to their ability to jointly process images, videos, and natural language queries, making them far more applicable in real-world scenarios than traditional unimodal retrievers. In particular, state-of-the-art MLLM-based rerankers harness powerful vision encoders and causal attention mechanisms to effectively fuse visual and textual information, usually ultimately regressing a single relevance score, and achieves strong performance in cross-modal matching tasks such as image search and e-commerce recommendation. However, despite these advances, outputting only a scalar relevance score leads to substantial information loss, as fine-grained semantic distinctions in complex user queries—such as compositional intent, attribute-level preferences, or contextual references—are inevitably collapsed into a single value. As an example, Fig. 1 illustrates a recommendation failure of the state-of-the-art (SOTA) multimodal reranker, Jina-Reranker-M0 [2], which prioritizes superficial semantic overlap between the user's query and the pink dress, resulting in an inappropriately high ranking for this item. Therefore, achieving accurate and fine-grained understanding of intricate multimodal queries has become an urgent challenge that must be addressed to realize the full potential of next-generation information matching systems.

Figure 1. An example illustrating **a limitation of Jina-Reranker-M0**. The user seeks a midi or long dress in a non-pink color for an evening event, yet the model ranks a short pink dress higher than a long black dress that is well-suited for such an occasion.

Recent studies on matching can be grouped into four categories. *1)* Representation-based architectures, as illustrated in Fig. 2a, independently embed the query and the document and then compute their similarity [6, 8, 15,

*Work done during an internship at Alibaba.
†Project Leader.
‡Corresponding Author.



(a) Representation-based Model (b) Adapted MLLM for Scoring (c) Autoregressive Model (d) Ours: YOFO

Figure 2. **Comparisons on Different Information Matching Methods.** (a) embeds inputs and then calculate similarity. (b) adapts an MLLM to output a single relevance score. (c) reformulates the matching problem as an autoregressive task, outputing relevance analysis. (d) Our method first decompose the text into a set of fundamental requirements and prefills them into a predefined template. Then the MLLM takes the template and the image as input and output whether each requirement is satisfied after a single forward pass.

26, 28, 30, 33]. However, fixed-size embeddings limit these models' capacity, making it difficult to capture all relevant information and to meet fine-grained requirements [35]. *2)* Interaction-based models have emerged to mitigate these limitations: they first compute low-level interaction representations between the query and document, and then a neural network analyzes these patterns to output a scalar relevance score for the query–document pair [7, 9, 13, 25]. These approaches achieve higher ranking quality than representation-based methods but are computationally more expensive and remain constrained by representational dimensionality. *3)* MLLM/LLM-based architectures, adapted to output a single relevance score improve matching quality by leveraging the deep semantic understanding and long-context capacity of large language models [24]. However, this design is misaligned with the native generative objective of LLMs and undermines fine-grained input understanding. *4)* LLMs and MLLMs can serve as native judges in a paradigm termed generative matching [39, 40], which reframes the task as autoregressive generation aligned with their generative nature. Nevertheless, these methods incur high computational overhead due to token-by-token decoding, making them difficult to deploy in real-world applications.

To address this dilemma, we observe that information matching essentially reduces to verifying whether a document satisfies a set of input requirements. We therefore propose **YOFO**, a single-forward-pass, template-conditioned judging paradigm that simultaneously determines, for each requirement in a predefined template, whether it is satisfied (yes/no). Specifically, YOFO adapts an autoregressive model to accept a structured requirement template and, after a single forward pass, reads the logits of the final token associated with each requirement; the induced next-token distribution discriminates between yes and no, enabling binary decisions without autoregressive decoding. Consequently, the model produces all requirement judgments in one forward pass, improving throughput while preserving interpretability. During training, YOFO follows standard next-token prediction but applies supervision only to positions that predict yes/no or reasoning tokens, encouraging precise and well-grounded judgments. At inference time, we first obtain a template by prompting an LLM to decompose the user's query into a set of requirements formatted according to a predefined schema (e.g., for "a blue, hooded, long-sleeve top without a chest logo," the requirements are: blue, hooded, long-sleeved, no chest logo). YOFO then consumes the template and, in a single forward pass, reads the final-token logits for each requirement and outputs a yes/no decision by applying softmax and selecting the higher-probability token between "yes" and "no". The resulting judgments can be post-processed as needed—for example, mapped to a relevance score using a human-defined mapping.

To evaluate the efficacy of our method, we derive a training set from SA-1B [16] and train the model on it to improve generalizability. We then construct a test set from LRVS-Fashion [17] to assess recommendation performance in a different domain. Extensive experiments demonstrate that our method accurately produces judgments conditioned on the input and template, and that later judgments can condition on earlier ones, enabling dependency-aware analysis. Furthermore, YOFO also benefits from post-hoc Chain-of-Thought (CoT), which encourages the model to reason implicitly before judging.

In summary, our contributions are as follows: (1) We introduce YOFO, an efficient compositional judging paradigm that judges a series of requirements concurrently in a single forward pass. (2) We demonstrate that YOFO enables dependency-aware analysis by conditioning later judgments on earlier ones. (3) We show that YOFO benefits from post-hoc CoT. (4) We further show that YOFO achieves state-of-the-art performance on recommendation tasks compared with traditional methods that output a sin-

gle relevance score, while preserving high throughput.

2. Related Works

2.1. Multimodal Large Language Models

Multimodal Large Language Models (MLLMs) [4, 5, 12, 20–23, 34, 36] have significantly extended the capabilities of Large Language Models (LLMs) [1, 3, 10, 18, 19, 31, 38] by enabling them to perceive, reason over, and generate responses grounded in multimodal inputs. However, the dense visual token sequences required for such alignment incur prohibitive computational costs. Recent approaches address this by either compressing tokens while preserving fidelity or rethinking how visual information is integrated. For instance, CrossLMM [37] employs dual cross-attention mechanisms to drastically reduce token count with minimal performance degradation. Similarly, MM-GEM [22] uses a PoolAggregator to support both generative and embedding tasks. These advances demonstrate that strategic token reduction can maintain multimodal performance at lower cost.

2.2. Rerankers

Reranking serves as a critical refinement stage in modern retrieval systems, aiming to re-order an initial candidate set by modeling fine-grained query–document interactions. Early neural rerankers employed representation-based architectures, calculating embeddings independently and then similarity, such as DSSM [8, 15], Convolutional DSSM [30]) and Long Short-Term Memory (LSTM) variants [6, 11, 26, 33]. But this kind of models are limited by the fixed-size embedding, and thus were soon superseded by interaction-based cross-encoders, most notably BERT-based models, that concatenate query and document as input and predict relevance via a classification or regression head over the [CLS] token. [25] first introduced this method and proved it effective. These models achieve strong performance but require pairwise inference, limiting scalability, and is constrained by fixed representational dimention and content length. With the rise of LLMs, researchers have repurposed powerful encoder-decoder (e.g., T5 [29]) or decoder-only (e.g., LLaMA [32]) backbones for reranking. A dominant paradigm [24, 41] fine-tunes LLMs to regress continuous relevance scores or classify relevance levels, leveraging their rich contextual understanding and long-context capabilities for improved semantic matching. Jina-Reranker-M0 also follows this paradigm but leverages Qwen2-VL [34], enabling support for multimodal inputs. More recently, generative reranking has emerged as a paradigm shift: instead of scoring candidates individually, the model directly generates the ranked sequence—such as a list of document indices or IDs—conditioned on the full candidate set. Methods like [39, 40] process all candidates jointly and autoregressively output the optimal ranking. This generative formulation naturally integrates reranking into end-to-end AI systems and shows particular promise in multimodal and agent-driven retrieval scenarios.

query \ documents	document 1	document 2
query	0.88	0.81

(a) Qeery: "I'm looking for some bottoms that are ideally suitable for summer wear. And I'd prefer them to be grey, if available.", document 1: "Grey formal trousers, a touch of warmth—perfect for winter.", document 2: "Black skirt, perfect for casual or date outfits in summer."

query \ documents	document 1	document 2
query	0.82	0.94

(b) Qeery: "I'm looking for a long-sleeve without a logo on the chest.", document 1: "A long-sleeve hooded.", document 2: "A long-sleeve with a fairly large logo on the chest."

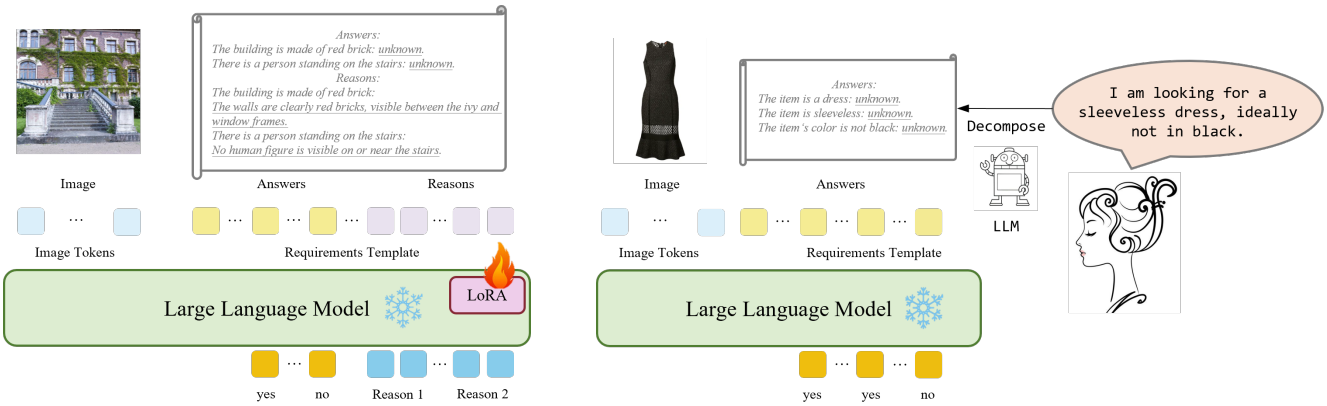
Table 1. Relevance scores computed by jina-reranker-m0 on text-to-text retrieval examples.

3. Method

In this section, we introduce YOFO, an efficient judging paradigm adapting an autoregressive model to receive a series of requirements as input and yields the judgements ("yes" or "no") for all requirements, respectively, as output. We begin by formulating the problem in Sec. 3.1, followed by the overall architecture of YOFO in Sec. 3.2, training in Sec. 3.3, inference in Sec. 3.4.

3.1. Problem formulation

As illustrated in Sec. 1, models akin to Jina-Reranker-M0 [2] struggle to perform accurate cross-modal retrieval. One might attribute this to visual information degradation caused by the image encoder and the connector projection, which impairs fine-grained alignment. However, we empirically find that these models also fail to retrieve text accurately from natural-language queries. As shown in Tab. 1a, Jina-Reranker-M0 assigns a higher score to winter trousers simply because they are gray, failing to recognize that the query explicitly requires summer-appropriate clothing, while "gray" is only a conditional preference. After further simplifying the prompt, Tab. 1b shows that Jina-Reranker-M0 fails to handle negation reliably, appearing to measure lexical overlap rather than semantic similarity on slightly more complex inputs. Verifying each requirement in the query individually would likely yield better results. To this end, we formally define the **compositional judging problem**.



(a) **YOFO Training** (b) **YOFO Inference**

Figure 3. **Architecture of YOFO.** (Left): YOFO uses an MLLM as its backbone. During training, the MLLM receives a structured requirement template as input and is trained to (i) judge each requirement and (ii) produce supporting reasons. YOFO supervises the model to output correct yes/no judgments while applying next-token prediction only to the reasoning text. (Right): At inference time, the user's query is first decomposed into a structured template, and the MLLM ingests the image and the template to determine, in a single forward pass, whether each requirement is satisfied, without autoregressively generating reasons.

The input to YOFO consists of an image $I \in \mathbb{R}^{3\times H\times W}$ and N requirements $(\boldsymbol{p}_i)_{i=1}^N$, for which the model determines whether they hold of the image. YOFO is a function that judges whether each requirement is satisfied with respect to the image:

$$f(I, (\boldsymbol{p}_i)_{i=1}^N) = (\boldsymbol{a}_i)_{i=1}^N \tag{1}$$

That is, YOFO judges each requirement $\boldsymbol{p}_i$ and outputs a binary answer $\boldsymbol{a}_i \in \{\text{yes}, \text{no}\}$.

3.2. Overall Architecture

YOFO leverages a decoder-only MLLM as the backbone. As illustrated in Fig. 3, we add a single token `unknown` to the end of each requirement $\boldsymbol{p}_i$, and concatenate all requirements to form the template $\boldsymbol{t}$. The template $\boldsymbol{t}$ is then tokenized into $\boldsymbol{s}$, and the indices of the single token `unknown` in the sequence are denoted $(pos_i)_{i=1}^N$. After a single forward pass, the model must predict whether each `unknown` position should be "yes" or "no". The forward process can be formulated as:

$$\boldsymbol{h} = \text{VLM}(I, \boldsymbol{t}) \tag{2}$$

The acquired logits $\boldsymbol{h}$ indicate the next-token distribution at each position. Therefore, for each requirement $\boldsymbol{p}_i$, the logits at position $pos_i - 1$ imply the distribution over the next token at pos_i, distinguishing "yes" from "no". In other words, it is this logit that we need to predict whether the requirement $\boldsymbol{p}_i$ is satisfied for the image input I. We read out the required logits as:

$$\boldsymbol{l}_i = \boldsymbol{h}[pos_i - 1], \quad i = 1, 2, \cdots, N \tag{3}$$

After that, these logits $\boldsymbol{l}_i \in \mathbb{R}^V$, where V is the vocabulary size of the LLM, are converted into a probability distribution via the softmax function, and the index of the maximum probability yields the model's prediction:

$$\boldsymbol{a}'_i = \text{vocab}[\arg\max(S(\boldsymbol{l}_i))], \quad i = 1, 2, \cdots, N \tag{4}$$

where S denotes softmax function. In real-world applications, to ensure that the LLM outputs binary results ("yes"/"no"), we can compare the probabilities of emitting "yes" and "no", and select the higher one as the final judgment. Therefore, this can be framed as below:

$$\boldsymbol{a}'_i = \begin{cases} \text{"yes"}, & \text{if } P(\text{"yes"}|s_{<pos_i}) > P(\text{"no"}|s_{<pos_i}) \\ \text{"no"}, & \text{otherwise} \end{cases} \tag{5}$$

where $i = 1, 2, \cdots, N$. Moreover, our experiments empirically reveal that, after training, the LLM learns to output only "yes" and "no"—even without explicit binarization.

3.3. Training

In this section, we describe YOFO's training procedure in detail. We first present the version without post-hoc Chain-of-Thought (CoT), and then introduce a variant with post-hoc CoT. The only difference is that, in the non–post-hoc CoT setting, the template omits the field of reasons that follow the answers.

Without Post-hoc CoT. As shown in Fig. 3a, we apply supervision at the answer positions to train the LLM to judge the requirements correctly. Specifically, we minimize the cross-entropy loss between the model logits $(\boldsymbol{l}_i)_{i=1}^N$ and the

ground-truth labels. The answer loss $\mathcal{L}_{\text{answer}}$ is defined as:

$$\mathcal{L}_{\text{answer}} = -\frac{1}{N}\sum_{i=1}^{N}\log P(a_i|s_{<pos_i}) \tag{6}$$

Although this objective resembles the standard autoregressive loss, when the model computes $\boldsymbol{l}_i$ for the i-th requirement, it does not observe the answers to previous requirements. Therefore, $\mathcal{L}_{\text{answer}}$ fundamentally differs from the autoregressive objective. Optimizing $\mathcal{L}_{\text{answer}}$ enables the LLM to make dependency-aware judgments, allowing later decisions to condition on its earlier judgements, as shown in Sec. 4.5.

With Post-hoc CoT. As shown in Fig. 3a, YOFO with Post-hoc CoT applies supervision to both the answers and the accompanying rationales. The requirements themselves are not supervised, as they are provided as input rather than predicted. Let $\mathcal{V}$ denote the positions of tokens in the reason field, that is $\mathcal{V} = \{i|s_i \text{ lies in the reason field}\}$. For each $i \in \mathcal{V}$, the logits at the preceding position $(i-1)$ predict the distribution over the next token s_i in the reason field. Accordingly, we define the following autoregressive loss to encourage the model to consider rationales prior to making judgments:

$$\mathcal{L}_{\text{reason}} = -\frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}}\log P(s_i|s_{<i}) \tag{7}$$

Because reason tokens substantially outnumber answer tokens and $\mathcal{L}_{\text{reason}}$ is introduced primarily to aid judgment, we scale $\mathcal{L}_{\text{reason}}$ by a coefficient λ, and define the total loss as:

$$\mathcal{L} = \mathcal{L}_{\text{answer}} + \lambda\mathcal{L}_{\text{reason}} \tag{8}$$

3.4. Inference

Fig. 3b illustrates how a YOFO-trained LLM performs inference. At inference time, there are no ground-truth annotations for rationales; consequently, the rationale field is omitted from the template. The MLLM backbone ingests the image I and the template $\boldsymbol{t}$ and performs a single forward pass, after which the logits $(\boldsymbol{l}_i)_{i=1}^{N}$ can be extracted as in Eq. (3). Finally, we apply the binary decision rule in Eq. (5) to ensure that the LLM outputs only "yes" or "no". The resulting judgments can then be consumed by downstream tasks—for example, provided as input to an agent for further analysis or mapped to a single score under a predefined criterion.

As an illustrative use case of YOFO, Fig. 3b depicts how it can be integrated into rerankers. To obtain a structured template, an LLM first decomposes the user's query into a set of requirements. These requirements are then assembled into a template and passed, together with the image, to the MLLM. Finally, all requirements are judged in a single forward pass, yielding results that substantially improve reranking quality. Task-specific rules can be devised to exploit these judgments; we leave the design open to practitioners.

4. Experiments

In this section, we present extensive experiments and ablation studies to validate our approach. We first compare YOFO with Jina-Reranker-M0 [2], a representative state-of-the-art (SOTA) multimodal reranker, on the test set derived from LAION-RVS-Fashion [17]. We then assess YOFO's dependency-aware judgement, the effectiveness of post-hoc CoT, and the contribution of individual architectural components.

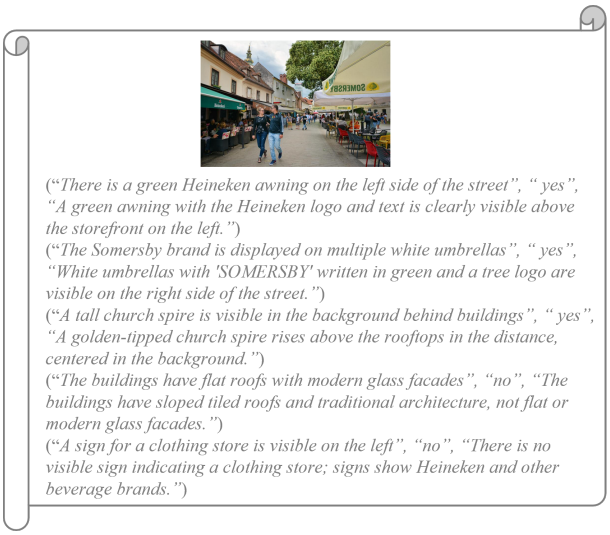
4.1. Impementation Details

We build on Qwen2-VL-2B-Instruct [34] and Qwen3-VL-2B-Instruct [27], freezing the vision encoder during training. We fine-tune the LLM using LoRA [14]. To conveniently locate the `unknown` token, we add four special tokens to the Qwen2-VL-2B-Instruct and Qwen3-VL-2B-Instruct processor: <|auth_start|>, <|auth_end|>, <|reason_start|>, and <|reason_end|>. The first two mark the start and end of the answer span, and the last two mark the boundaries of the reason span. Consequently, these special tokens enable straightforward label masking, allowing supervision to be applied exclusively to answers and reasons.

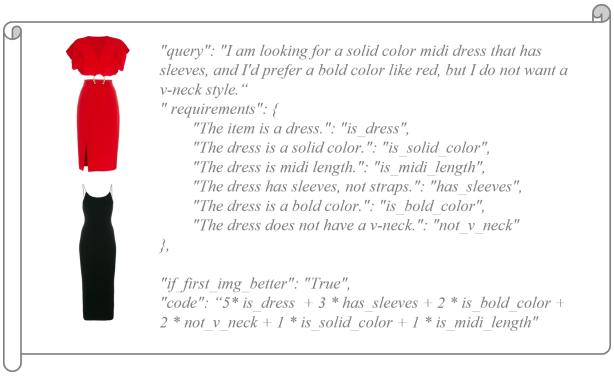
Training is conducted for one epoch with a learning rate of 1×10^{-4}, using the AdamW optimizer. We employ a cosine learning rate scheduler with a warmup ratio of 0.05. We set the maximum sequence length to 1,536 tokens. We use DeepSpeed ZeRO-3 for distributed training. We load the training data with eight worker processes, dropping the last incomplete batch. Training runs on eight H100 GPUs over four hours. We use bfloat16 mixed precision and enable gradient checkpointing to reduce memory usage. Moreover, we adopt $\lambda = 0.55$ as the default setting in all experiments using post-hoc CoT.

4.2. Data Construction

Training set. To construct the training dataset, we use the SA-1B dataset [16], introduced alongside the Segment Anything Model. It contains approximately 11M diverse, high-resolution images sourced from a privacy-preserving data engine and annotated with 1.1B high-quality segmentation masks. Since we do not require the masks, as shown in Fig. 4a, we use only the images, prompting an MLLM to analyze them, propose properties that hold or do not hold for the corresponding images, and provide reasons. The detailed prompt is shown in Fig. 5a. We randomly sample 1.2M images to ensure diversity and generalizability. For each image $\mathcal{I}_i$, the final sample is represented as $(\mathcal{I}_i, (\mathcal{P}_{ij}, \mathcal{A}_{ij}, \mathcal{R}_{ij})_{j=1}^{N})$, denoting the image, the proposed



(a) Training set example



(b) Test set example

Figure 4. **Examples of the training and test sets.** Each training sample consists of an image, a set of properties, answers, and corresponding reasons. Each test sample consists of two images, a customer-style query, the corresponding Python variables, ground-truth labels, and the expression used to compute each image's final recommendation score.

properties, the binary answers("yes" or "no"), and the reasons. To maintain a balanced and randomized mix of satisfied and unsatisfied properties, we prompt the MLLM to produce approximately equal numbers of each and shuffle them during training. Additionally, although we instruct the model to generate exactly ten properties per sample in the training set, the number of properties per training sample varies, typically remaining close to ten.. To evaluate the model and mitigate overfitting, we split the dataset at a ratio of $6:1$ into training and validation sets.

Test set. We use the LRVS-Fashion dataset [17], a large-scale public benchmark for referred visual search (RVS) in the fashion domain, to construct a test set. It comprises 272K fashion products and 842K images extracted from real-world catalogs. To evaluate generalization in Image-Text reranking, as depicted as Fig. 4b, we construct image pairs and queries; the model then judges which of the two images better matches the corresponding query. Specifically, we randomly sample image pairs $(\mathcal{I}_{i1}, \mathcal{I}_{i2})$ from the same category and, using an MLLM, generate a natural-language query $\mathcal{Q}_i$ that mimics how a customer would express their preferences. The MLLM then produces the answer $\mathcal{A}_i$ indicating whether the first image $\mathcal{I}_{i1}$ better matches $\mathcal{Q}_i$, the set of query requirements $(\mathcal{R}_{ij})_{j=1}^{M_i}$ (where M_i is the number of requirements), and a valid python expression $\mathcal{E}_i$ used to compute the final recommendation score based on our model's predictions. The detailed prompt is demonstrated in Fig. 5b.

4.3. Evaluation Metrics

In this section, we introduce the metrics used in our experiments to evaluate models.

On SA-1B validation sets and dependency-aware judgement task, we report two metrics to evaluate the accuracy of models. One metrics is Sample-wise Accuracy Acc_{sample}, denoting the percentage of samples on which the model judges all requirements correctly. Another metrics is Property-wise Accuracy Acc_{property}, indicating the percentage of properties judged correctly. Obviously, Acc_{sample} is always not greater than Acc_{property}.

To evaluate the model's ability to make judgements based on its previous ones, one more metrics is needed on dependency-aware judgement task, named Dependency-Aware Accuracy (Acc_{dep}).

On the test sets, we evaluate whether the model can rank the image pairs properly. Therefore, we report the error rate of ranking $Error_{\text{rank}}$, denoting the percentage of image pairs ranked wrongly.

4.4. Comparison on LAION-RVS-Fashion

In this section, we present an analysis of YOFO against state-of-the-art multimodal rerankers [2].

LAION-RVS-Fashion is a specialized fashion dataset, which inherently exhibits a significant domain shift relative to our training set constructed entirely on the broad and diverse SA-1B corpus. Despite this distributional gap, YOFO demonstrates remarkable robustness and adaptability when evaluated on fashion-specific recommendation tasks. As presented in Tab. 2, YOFO not only achieves strong performance but consistently outperforms the strong baseline jina-reranker-m0, even though it has never been exposed to fashion-related data during training. This result highlights a key advantage of our approach: YOFO learns general-purpose judging capabilities that transfer effectively across domains. Crucially, it can be deployed directly in specialized subdomains—such as fashion—without any fine-tuning or domain adaptation. This zero-shot generalization underscores the model's practical utility in real-world

Analyze the provided image and produce up to 10 concise declarative claims about visible objects, attributes, or spatial relations. Return them as a JSON object.

Requirements:
- Quantity and balance:
- Return exactly 10 key–value pairs, do not return 10 if you are uncertain about some of them.
- Exactly 5 keys must have the answer "yes" in the value and 5 keys must have the answer "no" in the value, as indicated by the "authenticity" field ("yes"/"no").
- Objectivity and verifiability:
- Claims must be precise, directly supported by the image, and free of speculation or subjective wording.
- Use the most specific common category you can state with high confidence (e.g., "dog" instead of "animal" if clearly visible), but do not over-specify.
- Coverage and diversity:
- Include a variety of attributes or relations
- Specificity and disambiguation:
- When multiple similar objects exist, disambiguate using descriptors.
- Language and style for keys:
- English, present tense, ≤15 words, no subjective adjectives, no quotes inside the key string.
- Uniqueness and non-triviality:
- Avoid redundant, overlapping, or trivially negated pairs (e.g., do not include both "There is a dog" and "There is no dog").
- Each key can describe a mixture of properties and relations.
- Constructing false claims:
- "no" authenticity claims should assert the absence of an object/attribute that is indeed not present or a relation that is not true, without contradicting your true claims.
- Reasoning field:
- For each claim, provide a brief reason citing specific visual evidence (e.g., colors, positions, relations, visible texts) and then answer whether the claim is true.
- Do not merely restate the claim; explain why it is true or false based on the image.

Output format:
- Return only a flat JSON object with up to 10 properties (no extra text).
- Each property's key is a single declarative claim string.
- Each property's value is an object with two fields:
- "authenticity": "yes" or "no" (lowercase only), indicating whether the claim is true or false, without contradicting your reason.
- "reason": a short English string explaining the visual evidence.
- Do not include additional fields, comments, or metadata.

(a) Prompt to construct the training set

You are a shopper who wants to buy one category of product but hasn't decided on a specific brand. You make a single, naturally phrased search request with some non-trivial logical structure (AND / OR / NOT combinations), then an AI product-search app returns two recommended images (the two images are shown above). Based only on those two images, reverse-engineer the most likely single search request the shopper made, list every attribute requirement implied by that request, judge whether the first image matches the search request better than the second, and give an expression used to calculate the matching score between the image the search request. Please produce a JSON result with four fields as specified below.

Requirements for your answer and strict output format:
- Return only one JSON object and nothing else (no explanatory text, no comments).
- The JSON object must contain exactly these keys: "query", "properties", "if_first_img_better", "code".
- "query" (string): a single, natural, fluent search request in English that contains a modestly complex logic combination (use AND / OR / NOT where appropriate). It should read like something a real shopper would say and should not include numeric sizes, prices, dates, or other concrete numbers, only generic attributes such as category, color, style, material are allowed.
- "properties" (object): a mapping where each key is a complete sentence describing an attribute of the subject in the query (english, present tense, ≤15 words, no subjective adjectives, no quotes), and each value is a short valid Python variable name that you choose to represent that attribute in scoring later. Include entries for every attribute used in the query.
- "if_first_img_better" (string): 'True' if the first recommended image is a better match for the search request than the second image, otherwise 'False'.
- "code" (string): a single Python expression (no assignment, no surrounding text) that computes a composite recommendation score from the variables you named in "properties". Decide and document weights implicitly via coefficients in the expression. Use only the variable names from "properties" and numeric constants. The expression should reflect the relative importance of attributes (higher-weight important attributes). You may use any valid python operator, but keep it a single expression.

Additional constraints on the contents:
1. The query must include logical combinations (AND / OR / NOT), but avoid overly convoluted nesting, just enough to express conditional preferences.
2. Attributes may include category, color, style, material type, presence/absence of hood or large logo, general fit (slim/loose), etc., but must not include numeric sizes, price ranges, or dates.
3. The search request should be practical, natural, and plausible for everyday shopping.
4. The search query should fully match the main subject of both images; however, beyond that, the query should not perfectly match either image, while strictly ensuring that one image is clearly more suitable for the query than the other.
5. When creating the scoring expression, choose weights that reflect the importance of each attribute in the query: required/very important attributes should have larger coefficients than soft/optional preferences.
6. The "properties" mapping must include every attribute term used in "query". Variable names must be valid Python identifiers.

Example:
- If the first image depicts gray formal trousers—ideal for adding a touch of warmth in winter—and the second shows a black skirt, perfect for casual or date-ready summer outfits, then the subjects of the two images are trousers and a skirt, respectively; therefore, the search query must focus on lower-body garments. The two key differences are that the pants in the first image are gray, while the skirt in the second image is summer-appropriate. These elements can be combined into a single search request, with a clear priority order specified. Therefore, then the query can be: "I'm looking for some bottoms that are ideally suitable for summer wear. And I'd prefer them to be grey, if available."

(b) Prompt to construct the test set

Figure 5. **Prompts for dataset construction.** (a) We prompt an MLLM to propose ten properties of a randomly selected image, with an equal number of satisfied and unsatisfied properties. (b) We prompt an MLLM to generate a customer-style query for an image pair, indicate whether the first image satisfies the query better than the second, decompose the query into a set of requirements, and produce an expression used to compute each image's final recommendation score during YOFO evaluation.

scenarios where labeled data is scarce or domain shifts are common, positioning YOFO as a versatile solution for cross-domain recommendation systems.

YOFO also delivers substantially higher throughput than

Methods	$Error_{\text{rank}}$ (%) ↓	Throughput (pairs/s) ↑
Jina-Reranker-m0	16.2	36.41
YOFO (Qwen2-VL)	4.8	35.08
YOFO (Qwen3-VL)	**3.7**	**47.6**

Table 2. **Ranking Error Rates on LAION-RVS-Fashion252 Dataset.** YOFO outperforms jina-reranker-m0, while maintaining high throughput.

generative rerankers. This efficiency derives from YOFO's single-forward-pass design: unlike generative or multi-stage rerankers [24, 25] that must process each requirement sequentially or in multiple rounds, YOFO produces all judgments in a single forward pass. Consequently, its inference latency increases only mildly with the number of requirements compared with generative or multi-stage rerankers, making it especially well suited to real-time, large-scale recommendation scenarios where both responsiveness and expressiveness are critical.

Additionally, the YOFO-trained Qwen3-VL-2B-Instruct model attains the lowest ranking error rate—1.1% lower than the model trained with Qwen2-VL-2B-Instruct—and achieves higher throughput, which we attribute to Qwen3-VL-2B's smaller parameter count.

4.5. Dependency-Aware Judgement

To evaluate whether our model can make judgments conditioned on previous ones, we randomly select two properties from each sample in the validation set and replace the second property with "**The answer to this question is the opposite of the answer to the previous question**". The model is therefore required to infer the answer to the second question without being given the first property's answer, while making both judgments simultaneously. We fine-tune the pretrained Qwen2-VL-2B-Instruct model on this dataset for one epoch, keep other settings unchanged, and then compare it with the model in Sec. 4.4. Although challenging, Tab. 3 demonstrates YOFO's dependency-aware judgment ability under the dependent-question setup, approaching 100%. Without training on the purposely-designed dataset, YOFO cannot judge correctly, because the properties in the training and validation sets can be judged individually, rendering dependency-aware analysis unnecessary. The base model performs poorly on this task, as it has not been exposed to scenarios where previous tokens are unobserved when predicting the next token.

4.6. Ablation Study

Effectiveness of YOFO training. We fine-tune Qwen2-VL-2B-Instruct and Qwen3-VL-2B-Instruct models on the training set using LoRA [14] for one epoch and evaluate it on the validation set. As shown in Tab. 4, after training, Property-wise Accuracy improves by over 20%, reaching above 90%, while Sample-wise Accuracy increases by approximately 40%. These ablation results demonstrate the effectiveness of our training paradigm. Moreover, the YOFO model trained with Qwen3-VL-2B-Instruct performs slightly better than its counterpart trained with Qwen2-VL-2B-Instruct. The Sample-wise Accuracy remains moderate because the training and validation sets contain, on average, nearly ten properties per sample, making it challenging to judge all properties correctly.

Methods	Acc_{dep} (%) ↑	Acc_{property} (%) ↑
Base	35.3	62.9
YOFO	57.6	71.4
YOFO + dep	**99.1**	**90.4**

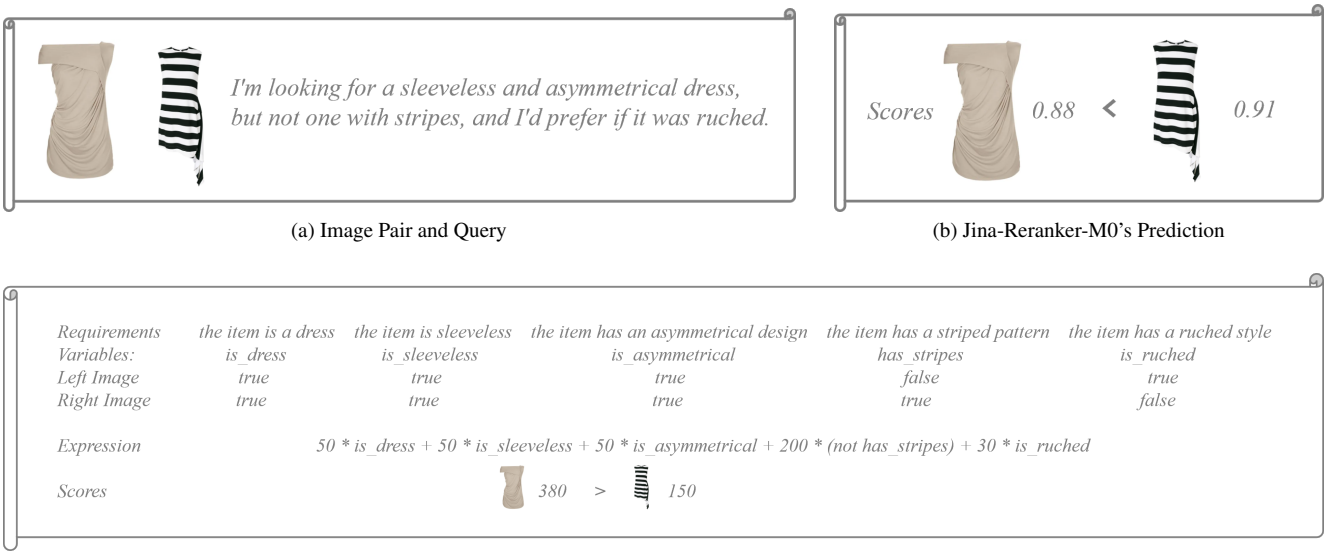
Table 3. **Accuracy on SA-1B validation set.** "Base"denotes pretrained Qwen2-VL-2B-Instruct model. "YOFO + dep" denotes YOFO trained with the dependency question. To construct questions that need dependency-aware analysis based on previous judgments, we randomly choose two properties from each sample in validation set, and replace the second requirement with "The answer to this question is the opposite of the answer to the previous question."

Methods	Acc_{property} (%) ↑	Acc_{sample} (%) ↑
Qwen2-VL		
Base	70.6	2.0
YOFO	91.2	41.8
YOFO + CoT	**91.3**	**42.0**
Qwen3-VL		
Base	68.4	1.3
YOFO	**92.3**	**46.6**

Table 4. **Ablation study of different components on SA-1B validation set.** We train models with Qwen2-VL-2B-Instruct and Qwen3-VL-2B-Instruct as base models. "YOFO" denotes the model obtained by fine-tuning the base model with YOFO. "YOFO + CoT" denotes the YOFO model augmented with post-hoc CoT.

Effectiveness of post-hoc CoT. As shown in the last row of Tab. 4, YOFO benefits from post-hoc CoT. This likely arises because the model is trained to generate and consider reasons before making judgments. This component is particularly useful in complex settings where detailed reasons can be annotated.

Rank of LoRA. We evaluate different ranks of LoRA, and the results in Tab. 5 show that $r = 16$ and $r = 64$ perform the best. To increase the model's representational capacity, we adopt $r = 64$ as our default experimental setting.



(a) Image Pair and Query

(b) Jina-Reranker-M0's Prediction

(c) YOFO's Prediction

Figure 6. **Illustrative comparison using a single example.** (a) presents a pair of images and a query. (b) shows that Jina-Reranker-M0 assigns a higher score to the second image; however, the dress in that image is striped and not ruched, so the prediction is incorrect. Panel (c) shows that after decomposing the query into five requirements, YOFO judges whether each requirement is satisfied by each image; these judgments are then combined using the specified expression to compute the final recommendation scores. Because all judgments are correct, the resulting prediction is accurate.

Rank	Acc_{property} (%) ↑	Acc_{sample} (%) ↑
32	91.2	41.7
64	**91.3**	**42.0**
128	91.2	41.7

Table 5. **Ablation study on LoRA rank using the SA-1B validation set.** As the LoRA rank increases, accuracy first rises and then declines, indicating that a rank of 64 yields the best performance.

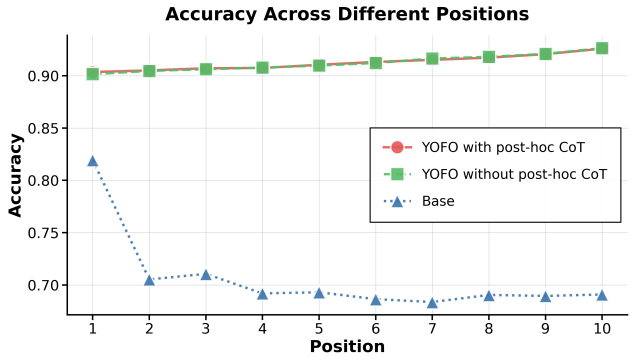


Figure 7. **The accuracy of different models across template positions on the SA-1B validation set.** "Base" denotes Qwen2-VL-2B-Instruct. As properties appear later in the sequence, YOFO maintains an accuracy of approximately 90%, while the base model shows a sharp decline in accuracy starting from the second property.

4.7. Additional Analysis

We present a case study in Fig. 6. The user seeks a sleeveless, asymmetric dress that is not striped and preferably has ruching. However, jina-reranker-m0 assigns a higher score to the dress in the second image, overlooking its stripes and lack of ruching. In contrast, YOFO judges, for each image, whether the query's requirements are satisfied and correctly assigns a higher score to the first image. This case explicitly demonstrates the effectiveness and interpretability of our method.

We also analyze judgment accuracy across positions on the SA-1B validation set. Specifically, Acc_i denotes the accuracy for i-th requirement, and Fig. 7 visualizes the results. The figure shows that the base model achieves over 80% accuracy on the first requirement, but its accuracy declines rapidly toward 70% thereafter. In contrast, YOFO maintains accuracy above 90%, with or without post-hoc CoT. This likely occurs because the base model's behavior on the first requirement resembles autoregressive generation, yet it fails to adapt to our template-conditioned judgment paradigm in which the ground-truth labels for previous judgments are unobserved. After training, YOFO adapts to this paradigm and consequently maintains high accuracy across positions.

5. Conclusions

In this work, we present You Only Forward Once (YOFO), a template-conditioned judging paradigm that judges whether

inputs satisfy requirements specified by a predefined template. By concurrently predicting next-token distributions for all requirements in a single forward pass, YOFO efficiently judges compositional requirements. Although trained on general-purpose data, YOFO achieves state-of-the-art performance on the reranking task and generalizes well across domains, enabling direct deployment in specialized settings. Our precise and efficient approach departs from traditional methods, which either lack fine-grained understanding of inputs or incur prohibitive computational overhead. Additionally, YOFO can condition later judgments on earlier ones and benefits from post-hoc CoT, making it effective in complex settings. Ultimately, our work establishes a new paradigm for judging, offering practical advantages in scenarios where multiple requirements must be assessed.

Limitations and Future Work. While our work demonstrates the effectiveness of YOFO in fine-grained multimodal compositional judging, it has a limitation that points to promising directions for future research. Specifically, we evaluate YOFO only on a single downstream task: reranking. Additional application scenarios warrant investigation. Future work could investigate additional applications of YOFO. Its explicit, interpretable judgments naturally lend themselves to serving as signals in reinforcement learning (RL) frameworks. Specifically, YOFO could be used as a structured reward model that provides fine-grained feedback (e.g., per-requirement rewards) rather than a single scalar score, thereby offering more precise guidance during RL training of MLLMs and diffusion models. Moreover, YOFO is well suited to multi-label classification and can therefore be applied to numerous scenarios, such as personalized recommendation systems that tag products and users' interests to deliver targeted recommendations.

References

[1] Imtiaz Ahmed, Sadman Islam, Partha Protim Datta, Imran Kabir, Naseef Ur Rahman Chowdhury, and Ahshanul Haque. Qwen 2.5: A comprehensive review of the leading resource-efficient llm with potentioal to surpass all competitors. *Authorea Preprints*, 2025. 1, 3

[2] Jina AI. jina-reranker-m0: Multilingual multimodal document reranker. https://jina.ai/news/jina-reranker-m0-multilingual-multimodal-document-reranker/, 2025. Accessed: 2025-11-13. 1, 3, 5, 6

[3] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023. 1, 3

[4] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization. *Text Reading, and Beyond*, 2:1, 2023. 1, 3

[5] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 1, 3

[6] Daniel Cohen and W Bruce Croft. End to end long short term memory networks for non-factoid question answering. In *Proceedings of the 2016 ACM international conference on the theory of information retrieval*, pages 143–146, 2016. 1, 3

[7] Zhuyun Dai and Jamie Callan. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 985–988, 2019. 2

[8] Jianfeng Gao, Patrick Pantel, Michael Gamon, Xiaodong He, and Li Deng. Modeling interestingness with deep neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 2–13, 2014. 1, 3

[9] Luyu Gao and Jamie Callan. Long document re-ranking with modular re-ranker. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2371–2376, 2022. 2

[10] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. 1, 3

[11] S Hochreiter and J Schmidhuber. Long short-term memory. *Neural Computation MIT-Press*, 1997. 3

[12] Jack Hong, Shilin Yan, Jiayin Cai, Xiaolong Jiang, Yao Hu, and Weidi Xie. Worldsense: Evaluating real-world omnimodal understanding for multimodal llms. *arXiv preprint arXiv:2502.04326*, 2025. 3

[13] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. *Advances in neural information processing systems*, 27, 2014. 2

[14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 5, 8

[15] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338, 2013. 1, 3

[16] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023. 2, 5

[17] Simon Lepage, Jérémie Mary, and David Picard. Lrvs-fashion: Extending visual search with referring instructions. *arXiv preprint arXiv:2306.02928*, 2023. 2, 5, 6

[18] Pengxiang Li, Shilin Yan, Joey Tsai, Renrui Zhang, Ruichuan An, Ziyu Guo, and Xiaowei Gao. Adaptive

classifier-free guidance via dynamic low-confidence masking. *arXiv preprint arXiv:2505.20199*, 2025. 3

[19] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024. 1, 3

[20] Shiyin Lu, Yang Li, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, and Han-Jia Ye. Ovis: Structural embedding alignment for multimodal large language model. *arXiv preprint arXiv:2405.20797*, 2024. 1, 3

[21] Shiyin Lu, Yang Li, Yu Xia, Yuwei Hu, Shanshan Zhao, Yanqing Ma, Zhichao Wei, Yinglun Li, Lunhao Duan, Jianshan Zhao, et al. Ovis2. 5 technical report. *arXiv preprint arXiv:2508.11737*, 2025. 1

[22] Feipeng Ma, Hongwei Xue, Guangting Wang, Yizhou Zhou, Fengyun Rao, Shilin Yan, Yueyi Zhang, Siying Wu, Mike Zheng Shou, and Xiaoyan Sun. Multi-modal generative embedding model. *arXiv preprint arXiv:2405.19333*, 2024. 3

[23] Feipeng Ma, Hongwei Xue, Yizhou Zhou, Guangting Wang, Fengyun Rao, Shilin Yan, Yueyi Zhang, Siying Wu, Mike Zheng Shou, and Xiaoyan Sun. Visual perception by large language model's weights. *Advances in Neural Information Processing Systems*, 37:28615–28635, 2024. 3

[24] Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2421–2425, 2024. 2, 3, 8

[25] Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. Multi-stage document ranking with bert. *arXiv preprint arXiv:1910.14424*, 2019. 2, 3, 8

[26] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM transactions on audio, speech, and language processing*, 24(4):694–707, 2016. 2, 3

[27] QwenTeam. Qwen3-vl: Sharper vision, deeper thought, broader action. https://qwen.ai/blog?id=99f0335c4ad9ff6153e517418d48535ab6d8afef&from=research.latest-advancements-list, 2025. Accessed: 2025-11-13. 5

[28] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 2

[29] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020. 3

[30] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd international conference on world wide web*, pages 373–374, 2014. 2, 3

[31] Qwen Team et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2(3), 2024. 1, 3

[32] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 3

[33] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. A deep architecture for semantic matching with multiple positional sentence representations. In *Proceedings of the AAAI conference on artificial intelligence*, 2016. 2, 3

[34] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. 1, 3, 5

[35] Zhichao Xu, Fengran Mo, Zhiqi Huang, Crystina Zhang, Puxuan Yu, Bei Wang, Jimmy Lin, and Vivek Srikumar. A survey of model architectures in information retrieval. *arXiv preprint arXiv:2502.14822*, 2025. 2

[36] Hongwei Xue, Yuchong Sun, Bei Liu, Jianlong Fu, Ruihua Song, Houqiang Li, and Jiebo Luo. Clip-vip: Adapting pre-trained image-text model to video-language alignment. In *The Eleventh International Conference on Learning Representations*, 2023. 3

[37] Shilin Yan, Jiaming Han, Joey Tsai, Hongwei Xue, Rongyao Fang, Lingyi Hong, Ziyu Guo, and Ray Zhang. Crosslmm: Decoupling long video sequences from lmms via dual cross-attention mechanisms. *arXiv preprint arXiv:2505.17020*, 2025. 3

[38] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 1, 3

[39] Eugene Yang, Andrew Yates, Kathryn Ricci, Orion Weller, Vivek Chari, Benjamin Van Durme, and Dawn Lawrie. Rank-k: Test-time reasoning for listwise reranking. *arXiv preprint arXiv:2505.14432*, 2025. 2, 3

[40] Soyoung Yoon, Eunbi Choi, Jiyeon Kim, Hyeongu Yun, Yireun Kim, and Seung-won Hwang. Listt5: Listwise reranking with fusion-in-decoder improves zero-shot retrieval. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2287–2308, 2024. 2, 3

[41] Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. Rankt5: Fine-tuning t5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2308–2313, 2023. 3