MORDAL: AUTOMATED PRETRAINED MODEL SELECTION FOR VISION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Incorporating multiple modalities into large language models (LLMs) is a powerful way to enhance their understanding of non-textual data, enabling them to perform multimodal tasks. Vision language models (VLMs) form the fastest growing category of multimodal models because of their many practical use cases, including in healthcare, robotics, and accessibility. Unfortunately, even though different VLMs in the literature demonstrate impressive visual capabilities in different benchmarks, they are handcrafted by human experts; there is no automated framework to create task-specific multimodal models.

We introduce Mordal, an automated multimodal model search framework that efficiently finds the best VLM for a user-defined task without manual intervention. Mordal achieves this both by reducing the number of candidates to consider during the search process and by minimizing the time required to evaluate each remaining candidate. Our evaluation shows that Mordal can find the best VLM for a given problem using $8.9\times$–$11.6\times$ lower GPU hours than grid search. We have also discovered that Mordal achieves $1.2\times$–$3.3\times$ better performance than the state-of-the-art model selection methods on a variety of tasks.

1 Introduction

Vision Language Models (VLMs) bridge the gap between visual and language understanding, rising as the dominant approach to solving visual information-based tasks. Notably, GPT-4o Hurst et al. (2024), a large-scale multimodal language model, demonstrates impressive vision reasoning capabilities by taking images as input and generating detailed natural language descriptions. Although the technical details behind GPT-4o remain undisclosed, researchers have proposed a number of publicly available VLMs (e.g., LLaVA Liu et al. (2023a), InternVL Chen et al. (2024b) and Qwen-VL Bai et al. (2023)) that aim to match GPT-4o's capabilities. Many of these open-source VLMs share a similar architecture, in which a feature projector converts the image embeddings generated by a vision encoder and feeds it to a large language model (LLM) along with the text embeddings.

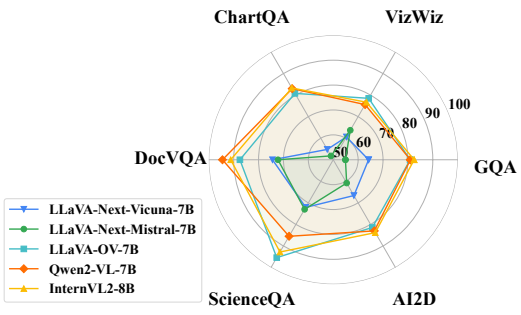


Figure 1: Benchmark performance of five latest *open-source* VLMs on six multimodal tasks.

To construct and train a VLM, the common approach starts from selecting an appropriate *pretrained* vision encoder and LLM. Thanks to the ever-growing ecosystem like HuggingFace Face (2025), developers are able to choose from countless pretrained models for their own VLMs. Unfortunately, despite the great number of available models, it is difficult to determine which pretrained models (i.e., vision encoders and LLMs) are the most appropriate ones. Given a user-specific downstream task, it is unclear which pretrained models can form the VLM that will meet the user's needs most effectively. As shown in Figure 1, no single VLM consistently outperforms the others in accuracy across all dimensions.

VLM capabilities vary significantly depending on their pretrained components Liu et al. (2023c); Xu et al. (2023); Zhang et al. (2024). It is unreliable and unpredictable to rely on human "intuitions"

to select pretrained models for the given downstream task, such as selecting the latest one or the most well-known one. Existing model selection methods, such as EMMS Meng et al. (2023) and LogME You et al. (2021), also fall short in the VLM context. These methods are primarily designed for classification, regression, or language-only tasks, where zero-shot performance can serve as an indicator of model quality Brown et al. (2020); Lin et al. (2024); Yi et al. (2024). However, VLMs require vision-text alignment, making zero-shot evaluations unreliable. Without proper alignment, the language model cannot correctly interpret the image embeddings, leading to random and meaningless outputs. With significant constraints on available time and computation cost, it is also unrealistic to try every pretrained model combination and train corresponding VLM candidates. Training a single VLM with vision-text alignment data could take more than 100 GPU hours Liu et al. (2023a); Karamcheti et al. (2024). This motivates the key question of this work: *How to effectively find the best pretrained models in a VLM given a downstream task?*

To address this question, we formulate the *pretrained model selection problem for VLMs* and model it as a resource-constrained task to predict the *alignment* performance of a VLM; i.e., the performance on the downstream task after training the feature projector with the vision-text alignment data. We empirically show that existing model selection methods fail to find the best pretrained model for downstream tasks and a naive approach like grid search is infeasible in practice.

We present **Mordal**, a novel pretrained model selection framework, which automatically and efficiently explores different pretrained model combinations in VLM. Mordal builds on our observation that efficiently solving this problem needs jointly considering two optimization directions: (1) minimizing the number of VLM candidates, where each candidate has different pretrained vision encoders and LLMs; and (2) reducing the evaluation time for each candidate. Overall, we make the following contributions in this work:

- We define the pretrained model selection problem in the context of VLMs and demonstrate that off-the-shelf VLMs often do not contain the best pretrained components for a given downstream task.
- We propose Mordal, an efficient pretrained model search framework, to find the best VLM for a given downstream task. Mordal clusters VLM candidates by their representation similarities while employing early stopping and scaling prediction mechanisms to reduce evaluation time.
- Extensive evaluations show that Mordal efficiently finds the best VLMs with $8.9\times$–$11.6\times$ less computation time than grid search. It also achieves $1.2\times$–$3.3\times$ better performance than the state-of-the-art model selection methods for a wide range of tasks.

2 Background and Motivation

We start by outlining the architecture of typical VLMs. Following this, we highlight the challenges in pretrained model selection for VLMs and show that existing VLMs often do not pick ideal pretrained models for downstream tasks. Based on these observations, we present the limitations of potential solutions like grid search, which motivate Mordal's design.

2.1 Vision Language Model

Common VLM architectures, like LLaVA Liu et al. (2023a), include a pretrained visual encoder to encode visual features, a pretrained large language model (LLM) to comprehend the user instructions and produce responses, and a vision-language cross-modal feature projector to align the vision encoder outputs to the language models:

Vision Encoder (VE). The vision encoder is responsible for processing input images and extracting relevant features. Potential encoder options are CLIP Radford et al. (2021), SigLIP Zhai et al. (2023), and InternViT Chen et al. (2024b), etc.

Feature Projector (FP). The vision-language cross-modal feature projector aims to align the encoded image features to the text token embedding space. The feature projector can be achieved directly by a Linear Projector or Multi-Layer Perceptron (MLP) Liu et al. (2023b).

Table 1: Evaluation results of capability on tasks of Visual QA, Doc QA, and Knowledge for four VLMs with different pretrained models. CLIP-Vicuna has the same pretrained models and mode structure as LLaVA-1.5-7B Liu et al. (2023a). The best result for each scenario is in **bold** text. There is no silver bullet.

Model	Vision Encoder	Language Model	Visual QA		Doc QA		Knowledge	
			GQA	VizWiz	ChartQA	DocVQA	ScienceQA	AI2D
CLIP-Vicuna	CLIP-ViT-L/14	Vicuna-1.5-7B	61.5	41.2	18.2	**27.6**	70.4	54.8
SigLIP-Vicuna	SigLIP-so400m-patch14	Vicuna-1.5-7B	**66.4**	**44.8**	**18.4**	24.1	68.5	53.0
CLIP-Llama	CLIP-ViT-L/14	Llama-3-8B	55.8	37.9	13.3	17.3	75.7	58.2
SigLIP-Llama	SigLIP-so400m-patch14	Llama-3-8B	56.4	38.1	13.4	17.4	**78.5**	**60.1**

Large Language Model (LLM). The language model processes mixed embeddings generated from both user instructions in text as well as image inputs. The commonly used LMs in VLMs are decoder-only LLMs, which include Llama Touvron et al. (2023a), Qwen Yang et al. (2024) and Mistral Jiang et al. (2023).

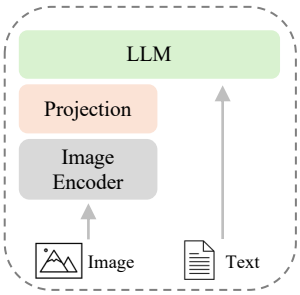


Figure 2: Overview of VLM architecture.

With this structure, a VLM first processes an input image $x_{img} \in \mathbb{R}^{H \times W}$ with an image processor and passes it to a vision encoder V. The vision encoder outputs a sequence of raw vision embeddings (or patches) $p_{img} \in \mathbb{R}^{L \times h_{vis}}$ where $p_{img} = V(x_{img})$ and h_{vis} is the hidden state dimension of the vision encoder outputs. The feature projector P will then map p_{img} to aligned vision embeddings $e_{img} \in \mathbb{R}^{L \times h_{text}}$ where h_{text} is the hidden dimension of the corresponding LM token. The aligned embeddings e_{img} will append to text prompt embedding $e_{text} = embed(prompt)$ and feed into the LLM to generate output text.

2.2 Pretrained Model Selection: No Silver Bullet

VLMs are versatile and powerful because most vision tasks can be formulated as next-token prediction. To train a VLM for a specific downstream task, developers usually *cherry-pick* pretrained vision encoders and language models for alignment. However, different pretrained models have varying capacities, which affect VLM performance. To investigate the impact of different pretrained models, we conduct grid search on 49 VLM candidates (i.e., seven vision encoders and seven language models) and train each candidate with *the same alignment data*. While complete evaluation is described in Section 4, we present the performance of four representative VLM candidates on six datasets, across three dimensions: Visual QA (GQA Hudson & Manning (2019) and VizWiz Gurari et al. (2018)), Doc QA (ChartQA Masry et al. (2022) and DocVQA Mathew et al. (2021)) and Knowledge (ScienceQA Lu et al. (2022) and AI2D Kembhavi et al. (2016)). As shown in Table 1, both pretrained vision encoders and language models have a significant impact on VLM performance. We further show in Appendix A that fine-tuning on specific datasets does not eliminate these differences, since model performance remains bounded by the quality of its pretrained components. Overall, *there is no silver bullet pretrained model or model combination that reigns supreme for all tasks*.

Given these observations, one may naturally ask: *how can we select the best combination of pretrained models for a specific task?* In this paper, we address the pretrained model selection problem in the context of VLMs. Given an alignment dataset and a target task, we aim to find the combination of a pretrained vision encoder and a language model that achieves the best performance on the target task after alignment.

2.3 Limitations of Existing Solutions

Solving the pretrained model selection problem by relying on empirical experiences or intuitions, such as choosing the newest, largest or most well-known model, is unreliable. For example, in Table 1, a VLM with Vicuna-1.5-7B can outperform a VLM with LLaMA-3-8B in certain settings. Existing model selection methods, such as EMMS Meng et al. (2023), LogME You et al. (2021), LEEP Nguyen et al. (2020a), and NLEEP Li et al. (2021), primarily focus on tasks involving classification or regression Brown et al. (2020). These methods typically assess models based on transferability metrics, which are inadequate for VLMs requiring alignment Vu et al. (2020); Yi et al. (2024); Lin et al. (2024). VLMs like LLaVA Liu et al. (2023a), as shown in Figure 2, involve

combining vision encoders with large language models (LLMs) through a feature projector to achieve multimodal interaction. Without alignment, the LLM cannot interpret image embeddings, rendering transferability metrics less effective.

Performing an exhaustive search for pretrained model selection is also computationally prohibitive. Even within a manageable search space, training a single VLM candidate with the LLaVA-1.5 dataset, using a pretrained 7B LLM backbone, can consume over 100 GPU hours. Considering the vast number of pretrained models available (e.g., over 150,000 LLMs on HuggingFace as of January 2025), exhaustively evaluating every candidate becomes impractical. The cost is further exacerbated as new encoders and LLMs continue to be introduced, necessitating constant re-evaluation to integrate new components.

To address the inefficiencies of exhaustive search in pretrained model selection, we must reduce the search cost along two key dimensions: (1) reducing the number of candidates and (2) minimizing the time required to evaluate each candidate. By optimizing these two dimensions, the exhaustive search process can be replaced with a more efficient pipeline that balances time consumption and selection accuracy.

3 Mordal Design

This section details the core components of Mordal's design. The exhaustive search is expensive because it (1) needs to evaluate every candidate (large search space), and (2) needs to train each candidate with a full dataset to see its performance (high evaluation cost). Mordal reduces search space by clustering the candidates based on their similarity and by introducing a two-step inter- and intra-cluster evaluation (§3.1), and reduces evaluation cost of each candidate with early stopping and scaling prediction (§3.2).

3.1 Candidate Clustering

With the rapid increase in the number of pretrained models in popular model zoos (e.g., HuggingFace Face (2025)), evaluating every candidate combination is expensive. Based on prior observations that similar models tend to have similar performance Hu et al. (2023); Yu et al. (2024); Lai et al. (2023), Mordal clusters candidates and evaluates them in two steps: inter-cluster and intra-cluster, to reduce the search space.

Measuring similarity. Measuring the similarity of VLM candidates – without training projectors between vision encoders and language models – is challenging. Parameter similarity Lai et al. (2023), which has been used to measure similarity between model architectures, does not fully consider the data distribution pattern in the target task. Models with high parameter similarity could still show different performance on different tasks. Therefore, in Mordal, we consider *representation similarity* between VLM candidates, which depends on the target task.

Mordal employs centered kernel alignment (CKA) Kornblith et al. (2019) to evaluate the similarity of representations between two VLM model structures. CKA has been proven to be an effective tool for understanding and comparing the information encoded across different layers of neural networks. Formally, CKA operates on two datasets by analyzing their corresponding activation matrices. The CKA score is defined as:

$$\text{CKA}(K, L) = \frac{\text{HSIC}(K, L)}{\sqrt{\text{HSIC}(K, K) \cdot \text{HSIC}(L, L)}} \tag{1}$$

where K and L are the kernel matrices of activations of two models. HSIC is the Hilbert-Schmidt Independence Criterion (HSIC) defined as:

$$\text{HSIC}(K, L) = \text{Tr}(KHLH) \tag{2}$$

where Tr() is the trace of a matrix and H is the centering matrix $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$. CKA is particularly useful in this context for two reasons. First, CKA can compare representations with differing shapes generated by different pretrained models, a task where traditional metrics such as cosine similarity fail. Second, as vision representations are commonly projected through MLP layers, this transformation does not compromise CKA's properties, making it robust and well-suited for such evaluations.

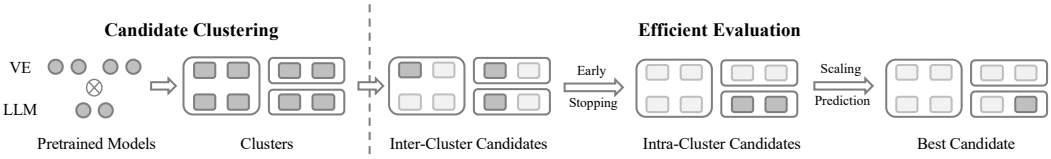
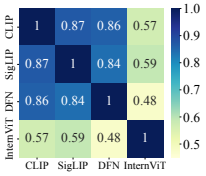


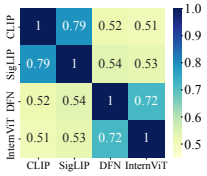
Figure 3: An overview figure for Mordal. Gray shapes represent pretrained models and VLM candidates, while white blocks are eliminated ones. Mordal clusters similar candidates, evaluates one per cluster, eliminates weak clusters, and uses regression to predict the best candidate.

Vision Encoder	ScienceQA	VizWiz
CLIP-ViT-L/14	67.6	41.2
SigLIP-so400m-patch14	67.7	44.8
DFN5B-CLIP-ViT-H/14	62.3	34.4
InternViT-300M	56.9	30.7

(a) Evaluation Results



(b) ScienceQA



(c) VizWiz

Figure 4: Evaluation results of VLMs with different vision encoders and the same language model (i.e., Vicuna-1.5-7B). Similarity scores between four vision encoders on ScienceQA and VizWiz.

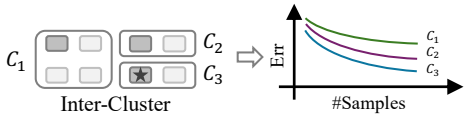
To validate the effectiveness of CKA, we train multiple VLM combinations with four vision encoders (CLIP [43], SigLIP [58], DFN-CLIP [12], InternViT [7]) and the same LLM backend Vicuna-1.5-7B. The trained VLMs are evaluated on two different datasets: ScienceQA and VizWiz. As shown in Figure 4, the image representation (i.e., outputs of projection heads) generated by different vision encoders show different levels of similarity to each other. For example, with input images from ScienceQA, CLIP, SigLIP, and DFN-CLIP generate similar representations. Meanwhile, DFN, CLIP and InternViT generate similar representations in VizWiz. From the results in Figure 4a, the vision encoders with similar representations will have similar performance with the same language model. Although this observation does not allow us to directly predict the target task's performance, it helps reduce the number of VLM candidates by eliminating similar pretrained models.

Two-step clustering. Computing the CKA score between each pair of candidates can be expensive, since K and L are activations from batched data input. To reduce the amount of pair-wise CKA computation, Mordal introduces a two-step VLM clustering strategy: (1) clustering vision encoders, (2) clustering language models based on a fixed vision representation. We detail the clustering process as follows (see full pseudocode in Appendix B):

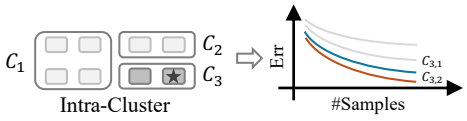
- *Vision encoder clustering.* Mordal computes the representation similarity between vision encoders using CKA. A distance matrix $Dist_{ve}$ is then constructed based on the dissimilarity values. The clustering function will take an input threshold t_{ve} and output the vision encoder clusters $\mathcal{C}_{ve}$.
- *Language model clustering.* Language model clusters are constructed based on vision representations of each vision encoder cluster. Using the medoid vision encoder from each cluster, Mordal generates a fixed image embedding for the dataset and a warmed-up feature projector transforms the shape of image embeddings to match the LLM input shape. A distance matrix $Dist_{llm}$ will record the dissimilarity and language models are then clustered based on an input threshold t_{llm}.

For each vision encoder cluster and the corresponding language model clusters, Mordal generates the candidate clusters by conducting the Cartesian product of the two clusters. The two-step clustering process reduces the computation costs by avoiding computing the similarity between candidates that have dissimilar vision encoders, which we show in Section 4.3 that usually yield distinct performance.

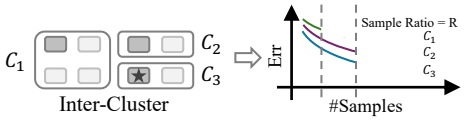
Inter- and intra-cluster evaluation. After grouping the candidates into clusters, Mordal finds the best candidate with two-step evaluation: inter-cluster evaluation and intra-cluster evaluation. The detailed process is shown in Figures 5a and 5b. Given that candidates in the same cluster have similar performance, inter-cluster evaluation first picks medoid from each cluster as the *representative*



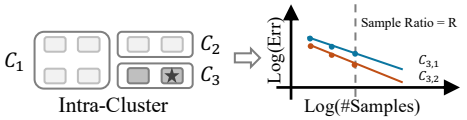
(a) Inter-cluster evaluation: selects top-K clusters. When $K = 1$, C_3 is selected.



(b) Intra-cluster evaluation: identifies the best candidate. Among the remaining two candidates, $C_{3,2}$ is selected.



(c) Inter-cluster evaluation with early stopping. Two candidates are early stopped and C_3 is selected.



(d) Intra-cluster evaluation with scaling prediction. $C_{3,2}$ is selected based on the predicted performance.

Figure 5: Inter- and intra-cluster evaluation with candidate clusters. Applying early stopping mechanism and scaling prediction to inter- and intra-cluster evaluation.

candidate and compares *performance of each cluster* to eliminate poorly performing clusters. The remaining Top-K clusters will be evaluated in the second step, where K is a user-defined parameter.

With the inter-cluster evaluation, many fewer candidates remain in consideration. Intra-cluster evaluation goes back to candidate-granularity evaluation by aggregating candidates from the remaining Top-K clusters. It trains all of them on the given alignment dataset, and returns the one with the best performance to the user.

3.2 Efficient Evaluation

Section 3.1 reduces the search space; however, evaluating each candidate in both stages (i.e., inter- and intra-) still requires training each candidate with a full dataset, which remains expensive. To address this, Mordal integrates two complementary techniques: early stopping for inter-cluster pruning, and scaling prediction for efficient intra-cluster evaluation. These components are co-designed to reuse intermediate checkpoints, further reducing redundant computation.

Early stopping. Mordal adopts the Successive Halving Algorithm (SHA) Jamieson & Talwalkar (2016) to aggressively eliminate low-quality candidates during inter-cluster evaluation. As shown in Figure 5c, SHA is conducted during inter-cluster evaluation, where all representative candidates are evaluated with the maximum data sample ratio R. It consists of multiple rounds, also known as *rung*. In each round, Mordal allocates a budget b to each candidate, evaluates all of them, and keeps the top $1/\eta$ candidates. In the next round, Mordal increases the budget to $b \times \eta$ per candidate. This repeats until representative candidates are converged or Top-K candidates are determined. In cases where the number of remaining candidates is large, Mordal also applies SHA again to intra-cluster evaluation. With SHA as the early stopping mechanism, Mordal *eliminates* poor-performing candidates earlier. This mechanism concentrates resources on the most promising clusters and produces intermediate checkpoints that can be reused downstream.

Scaling prediction. Scaling laws have traditionally been used to characterize the relationship between model size, training data, and performance in language models Kaplan et al. (2020); Hoffmann et al. (2022). These laws are typically formulated as a power-law relationship between LLMs' cross-entropy loss L and their compute scale measures, which take the form:

$$L(N, D) = \frac{a}{N^{\alpha}} + \frac{b}{D^{\beta}} + e \tag{3}$$

where N the number of model parameters, and D the number of training samples. The fitted parameters (α, β, a, b, e) are derived from training models across different scales. These relationships are typically log-linear under a log-log transformation and have been exploited for model selection in LLMs Lin et al. (2024); Ruan et al. (2024). In contrast to prior work, our focus is on alignment performance in VLMs, where the model size N is fixed, but the alignment data size D is varied. We find that a similar log-linear trend exists between the size of alignment data and task-specific error

metrics (details are provided in Section 4.3). This observational scaling law enables us to estimate a candidate VLM's full-data performance from a small number of low-cost training runs.

With the observational scaling law, Mordal employs a scaling prediction algorithm that automatically detects the log-linear scaling with sampled alignment dataset to conserve resources. As shown in Algorithm 1, for each candidate c in remaining candidate list C, the algorithm starts from an initial data sample ratio R (e.g., $\frac{1}{8}$). It evaluates a checkpoint trained on randomly sampled data with ratio R. The corresponding performance point $(Log(r), Log(Err))$ will be recorded in list P. After that, the algorithm reduces data sample ratio and repeats the above process until enough performance points (i.e., p) are collected and the log-linear relationship is observed for a given candidate c. Since data sample ratio r is decreasing, we may effectively start the evaluation of r/u from existing intermediate checkpoints to save computation costs.

Algorithm 1: Scaling Prediction

Input : Maximum data sample ratio R, scaling ratio u, minimum required point p, VLM candidate c
Output : Prediction result
Initialize loss-size pair list $P = []$
Initialize data sample ratio $r = R$
while *True* **do**
  /* Evaluate performance*/
  $Err = Evaluate(\mathcal{D}_{align}, \mathcal{D}_{task}, c, r)$
  $P.append((log(r), log(Err)))$
  if $|P| > p$ **then**
    Fit a linear regression model f_c on P
    Break if f_c fitting loss $> \delta$
  end
  $r = r/u$ /* Reduce data samples*/
end
Return $f_c(1)$

Scaling prediction complements Mordal's two-stage evaluation and helps reduce search cost. By fitting a log-linear model f_c on a few sampled alignment points P, Mordal can estimate the full-data performance $f_c(1)$ without fully training the candidate c. This enables early elimination of weaker candidates while prioritizing promising ones. Since the evaluation proceeds from larger to smaller sample ratios, intermediate checkpoints can be reused to save additional GPU time.

4 Evaluation

We conducted extensive experiments to thoroughly evaluate Mordal's performance. These experiments assessed its effectiveness in pretrained model selection and performed an ablation study. The key findings are summarized as follows:

1. Mordal identifies the best combination of vision encoder and LLM for the target task 8.9×-11.6× faster than the grid search. It also achieves 1.2×–3.3× better performance than the state-of-the-art model selection methods for a wide range of tasks.
2. We further validate the effectiveness of observational scaling law. By conducting the ablation studies, we show that each component in Mordal helps reduce the total search time while ensuring that it can still identify the top-performing candidates.

4.1 Experimental Setup

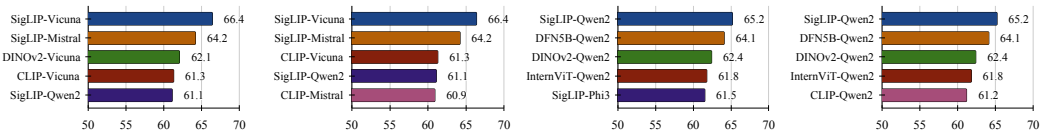
The experiments are conducted on seven mainstream datasets across three domains with seven vision encoders and seven LLMs. We deployed Mordal on a set of VMs on a cluster with a total of 16 NVIDIA A40 GPUs. Each GPU has 48 GB GDDR6 memory.

Dataset. We use LLaVA-1.5-Instruction dataset Liu et al. (2023a) for alignment. In practice, users may use their own alignment datasets. For target evaluation, we first select six standard benchmarks across Visual QA, Doc QA, and Knowledge tasks. To test broader generality, we further include MMMU, a multi-disciplinary benchmark covering diverse real-world domains. These datasets are commonly used to assess VLM performance Zhang et al. (2024).

Model zoo. We select seven vision encoders and seven language models based on popularity and performance. The selected vision models include both *language-supervised models* (e.g., OpenAI CLIP) and *self-supervised models* (e.g., DINOv2). For LLMs, all models follow the decoder-only structure, and we pick the most commonly used 7B LLMs from *HuggingFace*. The vision encoder is connected to the LLM using a two-layer MLP projector. The complete list of models and additional training details are provided in Appendix D.

Table 2: Summary of improvements. Search Time (h) and Top-1 Model Quality (%) results for different datasets. For each task, grid search exhaustively evaluates 49 candidates, requiring *5439 GPU hours*. Mordal significantly reduces the amount of training needed (i.e., GPU Saving) while successfully finding the Top-1 VLM candidate.

Task	Dataset	CLIP-Vicuna (LLaVA-1.5)	**Grid Search**		**Mordal (ours)**		
			Top-1 Score	Model Name	Time	Speedup	Top-1 Score
Visual QA	GQA Hudson & Manning (2019)	61.5	66.4	SigLIP-Vicuna	483	11.2×	**66.4**
	VizWiz Gurari et al. (2018)	41.2	46.9	SigLIP-Mistral	469	11.6×	**46.9**
Doc QA	ChartQA Masry et al. (2022)	18.2	**20.2**	CLIP-Qwen	607	8.9×	18.6
	DocVQA Mathew et al. (2021)	27.6	28.5	SigLIP-Qwen	593	9.2×	**28.5**
Knowledge	ScienceQA Lu et al. (2022)	70.4	78.5	SigLIP-Llama	472	11.5×	**78.5**
	AI2D Kembhavi et al. (2016)	54.8	65.2	SigLIP-Qwen	496	10.9×	**65.2**
Multi-discipline	MMMU Yue et al. (2024)	35.3	36.6	SigLIP-Llama	503	10.8×	**36.6**



(a) Grid Search (GQA) (b) Mordal (GQA) (c) Grid Search (AI2D) (d) Mordal (AI2D)

Figure 6: Top-5 candidates from grid search and Mordal on GQA and AI2D.

Baseline and metric. We use grid search as the primary baseline, where every VLM candidate is fully trained and evaluated to find the top-1 model. We also measure the total training time, in GPU hours, required by both grid search and Mordal. To evaluate ranking quality, we compare candidate rankings from grid search (based on performance) and Mordal (based on elimination order) using weighted Kendall's τ coefficient. We further compare Mordal with four model selection baselines: EMMS Meng et al. (2023), LogME You et al. (2021), LEEP Nguyen et al. (2020a), and NLEEP Li et al. (2021). For a fair comparison, we use the same alignment data and ensure that the total training time allocated to each candidate matches Mordal's overall budget. This allows us to assess both accuracy and efficiency under equal resource constraints. Implementation details can be found in Appendix D.

4.2 Performance Results

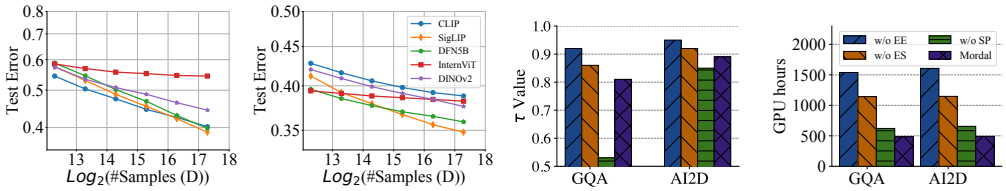
This sections evaluates Mordal's effectiveness in identifying top-performing VLM candidates under constrained training budgets. We also report the weighted Kendall's τ between Mordal's candidate rankings and baselines' candidate rankings to assess ranking quality.

Mordal is significantly faster than grid search. Table 2 shows that grid search requires 5439 GPU hours to exhaustively train 49 VLM candidates per task. In contrast, Mordal reduces the total search time by 8.9× to 11.6× across tasks, with search time varying by cluster formation and candidate convergence rates. Despite this reduction, Mordal successfully identifies the top-1 model in six out of seven tasks. For example, on VizWiz, Mordal completes the search in 469 GPU hours and selects SigLIP-Mistral, which achieves 46.9% accuracy. Notably, for most tasks, the best VLM candidates have different pretrained model combinations and all of them surpass the performance of the default LLaVA-1.5-7B structure, highlighting the importance of pretrained model selection.

Mordal outperforms existing model selecing methods in finding top-performing models. Mordal consistently outperforms existing model selection methods (EMMS, LogME, LEEP, and NLEEP) by achieving higher weighted Kendall's τ scores across all seven datasets, as shown in Table 3. For example, on the ScienceQA dataset, Mordal achieves a τ value of 0.96, significantly surpassing the best baseline (EMMS) with a τ of 0.77. This is because Mordal effectively captures the alignment performance of VLM combinations, while baseline methods rely on features from foundation models that do not fully distinguish between different model pairings. However, Mordal can occasionally miss the best combinations when promising candidates are grouped with weaker ones or are prematurely excluded by the early stopping mechanism (See Figure 6). Despite this limitation, Mordal consistently identifies top-performing models more accurately and efficiently than traditional methods.

Table 3: Comparison of different model selection modethods with the same time consumption. Kendall τ represents the differences of top-performing candidates compared with the groundtruth (i.e., grid search) and *larger τ is better*.

Task	Dataset	EMMS	LogME	LEEP	NLEEP	Mordal (Ours)
Visual QA	GQA Hudson & Manning (2019)	0.682	-0.162	0.232	0.435	**0.814**
	VizWiz Gurari et al. (2018)	0.657	0.236	0.351	0.502	**0.882**
Doc QA	ChartQA Masry et al. (2022)	0.238	-0.144	-0.071	0.298	**0.765**
	DocVQA Mathew et al. (2021)	0.172	0.155	0.111	0.265	**0.897**
Knowledge	ScienceQA Lu et al. (2022)	0.770	0.269	0.344	0.562	**0.960**
	AI2D Kembhavi et al. (2016)	0.557	0.193	0.316	0.614	**0.894**
Multi-discipline	MMMU Yue et al. (2024)	0.526	0.101	0.245	0.220	**0.875**



(a) GQA (b) AI2D (c) τ Value (d) Time

Figure 7: Observational scaling law validation and ablation study results for efficient evaluation (EE), early stopping (ES) and scaling prediction (SP). The results are on GQA and AI2D.

4.3 Ablation Studies

In this section, we first validate observational scaling law in Mordal. Then we conduct a comprehensive ablation study to evaluate the candidate clustering and efficient exploration in Mordal and their contributions to overall performance.

Observation scaling law validation. To validate the existence of scaling laws in VLM alignment, we train multiple VLM combinations with different vision encoders and the same language model Qwen2-7B on the sampled LLaVA-1.5-Instruction dataset. The trained VLMs are evaluated on two different datasets: GQA and AI2D. As shown in Figures 7a and 7b, we observe log-linear scaling across different VLMs, which supports the design of scaling prediction. However, the log-linear scaling will only appear after a certain number of training samples, which is consistent with the conclusion in previous work Lin et al. (2024); Ruan et al. (2024).

Effect of candidate clustering. Candidate clustering plays a vital role in Mordal as it enables inter- and intra-cluster evaluation. As illustrated in Figure 7d, inter- and intra-cluster evaluation (i.e., Mordal without efficient evaluation) significantly reduces training time while maintaining a high τ value. By grouping candidates with similar characteristics into clusters, Mordal evaluates representative candidates from each cluster first and eliminates candidates in poor-performed clusters.

Effect of efficient exploration. Early stopping mechanism prunes candidates during the early stage of training. While it significantly reduces the search time, applying it during the entire evaluation (i.e., Mordal without scaling prediction) will eliminate some promising candidates (e.g., SigLIP-Qwen on AI2D shown in Figure 7b). It evaluates candidates based on intermediate performance and leads to a low τ value. Mordal limits the usage of early stopping and introduces scaling prediction instead to predict the performance of promising candidates. As shown in Figure 7c, this leads to a significant improvement in the τ value while further reducing the total training time.

5 Conclusion

We presented Mordal, an automated framework for efficient pretrained model selection in vision-language models (VLMs). Mordal reduces search cost by pruning candidate combinations and minimizing evaluation time per model. Experiments show that Mordal achieves up to 11.6× speedup over grid search while maintaining top-1 accuracy, and outperforms existing baselines by 1.2×-3.3× in ranking quality. This highlights the effectiveness of task-aware, alignment-required model selection for VLM deployment.

Ethics Statement

This work adheres to the ICLR Code of Ethics. Our study does not involve human subjects or sensitive personal data. The datasets we use are publicly available and widely adopted in the research community. We do not anticipate any harmful societal impact from our methodology, and we believe that the contributions of this work will positively benefit the community.

Reproducibility Statement

We have taken steps to ensure reproducibility of our results. Detailed descriptions of datasets, experimental settings, and evaluation protocols are included in the main text and appendix. Model configurations, hyperparameters, and training details are provided. In addition, we will release the source code, model checkpoints, and full evaluation results in the future. We believe this will enable other researchers to fully reproduce and extend our work.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433, 2015.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37:27056–27087, 2024a.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24185–24198, 2024b.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL `https://lmsys.org/blog/2023-03-30-vicuna/`.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American chapter of the association for computational linguistics: human language technologies*, pp. 4171–4186, 2019.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Hugging Face. Hugging face models, 2025. URL `https://huggingface.co/models?sort=downloads`. Accessed: 2025-01-30.

Alex Fang, Albin Madappally Jose, Amit Jain, Ludwig Schmidt, Alexander T Toshev, and Vaishaal Shankar. Data filtering networks. In *International Conference on Learning Representations*, 2023.

Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3608–3617, 2018.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Qinghao Hu, Zhisheng Ye, Meng Zhang, Qiaoling Chen, Peng Sun, Yonggang Wen, and Tianwei Zhang. Hydro: Surrogate-based hyperparameter tuning service in datacenters. In *USENIX Symposium on Operating Systems Design and Implementation*, pp. 757–777, 2023.

Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6700–6709, 2019.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. URL `https://doi.org/10.5281/zenodo.5143773`. If you use this software, please cite it as below.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Artificial intelligence and statistics*, pp. 240–248. PMLR, 2016.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. *arXiv preprint arXiv:2402.07865*, 2024.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pp. 235–251. Springer, 2016.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pp. 3519–3529. PMLR, 2019.

Fan Lai, Yinwei Dai, Harsha V Madhyastha, and Mosharaf Chowdhury. Modelkeeper: Accelerating dnn training via automated training warmup. In *USENIX Symposium on Networked Systems Design and Implementation*, pp. 769–785, 2023.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

Yandong Li, Xuhui Jia, Ruoxin Sang, Yukun Zhu, Bradley Green, Liqiang Wang, and Boqing Gong. Ranking neural checkpoints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2663–2673, 2021.

Haowei Lin, Baizhou Huang, Haotian Ye, Qinyu Chen, Zihao Wang, Sujian Li, Jianzhu Ma, Xiaojun Wan, James Zou, and Yitao Liang. Selecting large language model to fine-tune via rectified scaling law. *arXiv preprint arXiv:2402.02314*, 2024.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023b.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. Mmbench: Is your multi-modal model an all-around player?, 2023c.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`, 2022.

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. *arXiv preprint arXiv:2203.10244*, 2022.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2200–2209, 2021.

Fanqing Meng, Wenqi Shao, Chonghe Jiang, Kaipeng Zhang, Yu Qiao, Ping Luo, et al. Foundation model is efficient multimodal multitask model selector. *Advances in Neural Information Processing Systems*, 36:33065–33094, 2023.

Cuong Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. Leep: A new measure to evaluate transferability of learned representations. In *International Conference on Machine Learning*, pp. 7294–7305. PMLR, 2020a.

Thao Nguyen, Maithra Raghu, and Simon Kornblith. Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth. *arXiv preprint arXiv:2010.15327*, 2020b.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

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*, pp. 8748–8763. PMLR, 2021.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128, 2021.

Yangjun Ruan, Chris J Maddison, and Tatsunori Hashimoto. Observational scaling laws and the predictability of language model performance. *arXiv preprint arXiv:2405.10938*, 2024.

Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint arXiv:2303.15389*, 2023.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *arXiv preprint arXiv:2406.16860*, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

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*, 2023b.

Anh T Tran, Cuong V Nguyen, and Tal Hassner. Transferability and hardness of supervised classification tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1395–1405, 2019.

Sebastiano Vigna. A weighted correlation index for rankings with ties. In *Proceedings of the 24th international conference on World Wide Web*, pp. 1166–1176, 2015.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. Exploring and predicting transferability across nlp tasks. *arXiv preprint arXiv:2005.00770*, 2020.

Jiayu Wang, Yifei Ming, Zhenmei Shi, Vibhav Vineet, Xin Wang, Sharon Li, and Neel Joshi. Is a picture worth a thousand words? delving into spatial reasoning for vision language models. *Advances in Neural Information Processing Systems*, 37:75392–75421, 2024.

Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Shengqiong Wu, Hao Fei, Leigang Qu, Wei Ji, and Tat-Seng Chua. Next-gpt: Any-to-any multimodal llm. In *International Conference on Machine Learning*, 2024.

Peng Xu, Wenqi Shao, Kaipeng Zhang, Peng Gao, Shuo Liu, Meng Lei, Fanqing Meng, Siyuan Huang, Yu Qiao, and Ping Luo. Lvlm-ehub: A comprehensive evaluation benchmark for large vision-language models. *arXiv preprint arXiv:2306.09265*, 2023.

Le Xue, Manli Shu, Anas Awadalla, Jun Wang, An Yan, Senthil Purushwalkam, Honglu Zhou, Viraj Prabhu, Yutong Dai, Michael S Ryoo, et al. xgen-mm (blip-3): A family of open large multimodal models. *arXiv preprint arXiv:2408.08872*, 2024.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong

Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Chao Yi, Yuhang He, De-Chuan Zhan, and Han-Jia Ye. Bridge the modality and capability gaps in vision-language model selection. *Advances in Neural Information Processing Systems*, 37: 34429–34452, 2024.

Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. Logme: Practical assessment of pre-trained models for transfer learning. In *International Conference on Machine Learning*, pp. 12133–12143. PMLR, 2021.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*, 2024.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9556–9567, 2024.

Xinyue Zeng, Haohui Wang, Junhong Lin, Jun Wu, Tyler Cody, and Dawei Zhou. Lensllm: Unveiling fine-tuning dynamics for llm selection. *arXiv preprint arXiv:2505.03793*, 2025.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11975–11986, 2023.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, et al. Lmms-eval: Reality check on the evaluation of large multimodal models. *arXiv preprint arXiv:2407.12772*, 2024.

Orr Zohar, Xiaohan Wang, Yann Dubois, Nikhil Mehta, Tong Xiao, Philippe Hansen-Estruch, Licheng Yu, Xiaofang Wang, Felix Juefei-Xu, Ning Zhang, et al. Apollo: An exploration of video understanding in large multimodal models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 18891–18901, 2025.

A Task-Specific Fine-Tuning Does Not Eliminate Performance Differences

Table 4: Task-specific fine-tuning results across three datasets.

Model	GQA	ChartQA	ScienceQA
CLIP-Vicuna	62.8	24.4	71.3
SigLIP-Vicuna	**67.4**	**26.3**	70.2
CLIP-LLaMA	56.5	22.9	78.1
SigLIP-LLaMA	58.7	23.2	**81.5**

An important question is whether performance differences across VLMs disappear after task-specific fine-tuning. Prior studies in both single-modality and multimodal settings indicate that this is not the case: performance differences often persist even after fine-tuning. For example, Lin et al. (2024) and Zeng et al. (2025) show that not all pretrained LLMs converge to similar performance after fine-tuning, even with identical data and compute budgets. Our work extends this investigation to the multimodal domain.

We conducted 12 task-specific fine-tuning runs: the four representative VLM combinations from Table 1 of the paper were fine-tuned on three target datasets that differ in their overlap with the LLaVA-1.5 alignment mixture—GQA (in-distribution), ChartQA, and ScienceQA (out-of-distribution). Results are shown in Table 4. Even after fine-tuning on data from each target distribution, we still observe performance differences of up to 10–11 percentage points among models. Notably, the top-1 model remains consistent with the selection reported in Table 1 of the paper. These findings indicate that Mordal generalizes well with the alignment data mixture and remains effective under distribution shift. Practically, this result highlights that even if a model is fine-tuned, its performance is bounded by the quality of its pretrained components. Thus, selecting strong pretrained VLMs remains crucial. When sufficient fine-tuning data is available, Mordal can also be applied directly on the target data to prune candidates, further reducing model selection time and compute.

B Algorithm

Algorithm 2: Candidate Clustering

Input : Target task D, Model Zoo $\mathcal{M}_{ve}$ and $\mathcal{M}_{llm}$, clustering threshold t_{ve} and t_{llm}
Output : Candidate clusters C_{vlm}

Def *CandidateClustering*($\mathcal{M}_{ve}, \mathcal{M}_{llm}$)**:**
  /* Vision Encoder Clustering*/
  for $M_A, M_B \in \mathcal{M}_{ve}$ **do**
    $dist_{A,B} = 1 - CKA(\mathcal{D}, M_A, M_B)$
    $Dist_{ve}[M_A][M_B] = Dist_{ve}[M_B][M_A] = dist_{A,B}$
  end
  $\mathcal{C}_{ve} = Clustering(Dist_{ve}, t_{ve})$
  /* Candidate Clustering*/
  for $C_A \in \mathcal{C}_{ve}$ **do**
    $M_{medoid} = PickMedoidModel(C_A)$
    /* LLM Clustering */
    for $M_A, M_B \in \mathcal{M}_{llm}$ **do**
      $dist_{A,B} = 1 - CKA(M_{medoid}(\mathcal{D}), M_A, M_B)$
      $Dist_{llm}[M_A][M_B] = Dist_{llm}[M_B][M_A] = dist_{A,B}$
    end
    $\mathcal{C}_{llm} = Clustering(Dist_{llm}, t_{llm})$
    $\mathcal{C}_{vlm}.append(C_A \times \mathcal{C}_{llm})$
  end
  Return $\mathcal{C}_{vlm}$

Detailed two-step clustering algorithm for candidate clustering as described in Section 3.1. We adopt the MinibatchCKA for computation efficiency, which is introduced in Nguyen et al. (2020b) and later used in Raghu et al. (2021). In LLM clustering, we use the last hidden state from LLM as the sentence representation for CKA computation since it produces the best clustering performance. We

leverage the hierarchical clustering from `scipy.cluster.hierarchy` library in SciPy Virtanen et al. (2020). It is possible to adopt other clustering methods. Overall, this approach significantly reduces the number of candidate to explore.

C Implementation

```python
import mordal

def search_with_mordal(model_zoo, alignment_data, target_task_data):
model = mordal.query_for_model(
data=alignment_data, # LLaVA-1.5-Mixture
task=target_task_data, # GQA
        pretrained_ve_zoo=model_zoo['ve'],
        pretrained_llm_zoo=model_zoo['llm'],
vlm_kwargs={
'projector': 'MLP', 'freeze_ve': True, 'freeze_llm': False,
        },
mordal_kwargs={
'clustering': {'t_ve': 0.7, 't_llm': 0.8},
'exploration': {'top_k_inter': 3, 'top_k_intra': 3},
'early_stopping': {'R': 0.125, 'b': 0.03, 'eta': 2}
'scaling_prediction': {'R': 0.125, 'u': 2, 'delta': 0.01}
        }
)
```

Listing 1: Mordal interface.

This section describes Mordal's implementation details and configurations to ensure efficient and scalable pretrained model selection. We highlight key design choices that optimize resource utilization without compromising performance. As shown in code snippet Listing 1, to submit a job, users need to provide the alignment data and data for the target task. Users are also allowed to submit a list of available pretrained models. In `vlm_kwargs`, users may specify the projector's architecture and whether to free pretrained components. When unfree pretrained components, instead of performing expensive full finetuning, we uses Low-Rank Adaptation (LoRA) Hu et al. (2021) implemented by Parameter-Efficient Fine-Tuning (PEFT) Mangrulkar et al. (2022) and manage LoRA configurations for each pretrained model. LoRA injects task-specific adaptations into the pretrained model by learning low-rank updates for certain layers while keeping the core parameters frozen. This significantly reduces computational and memory overhead, making finetuning feasible under resource-constrained settings. We incorporate Flash Attention Dao (2023) for scalable attention computation, which is a memory-efficient implementation of scaled dot-product attention that avoids redundant operations and reduces memory overhead. All models are trained with torch bfloat16 precision, which balances computational efficiency and numerical stability. Mordal also automatically allocates idle GPU resources to candidates that are not converged to speed up the exploration process.

D Additional Experiments

Vision Encoders	LLMs
CLIP-ViT-L/14@336 Radford et al. (2021)	Vicuna-1.5-7B Chiang et al. (2023)
SigLIP-so400m-patch14@384 Zhai et al. (2023)	Llama-2-7B Touvron et al. (2023b)
DFN-CLIP-ViT-H/14@378 Fang et al. (2023)	Llama-3-8B Dubey et al. (2024)
InternViT-300M/14@448 Chen et al. (2024b)	Mistral-v0.2-7B Jiang et al. (2023)
DINOv2-ViT-L/14@518 Oquab et al. (2023)	Qwen2-7B Yang et al. (2024)
EVA-CLIP-02-ViT-L/14@336 Sun et al. (2023)	Phi-3-Small-7B Abdin et al. (2024)
ConvNeXt-L/14@256 Ilharco et al. (2021)	Gemma-1.1-7B Team et al. (2024)

Table 5: List of vision encoders and LLMs in experiments.

Model zoo and training settings. We evaluate seven vision encoders and seven language models as shown in Table 5. Most models are available on *HuggingFace* Face (2025) while EVA-CLIP and

ConvNeXt are supported by *timm* library Wightman (2019). For ConvNeXt, we interpolate the output embeddings to 16x16 patches following Cambrian-1 Tong et al. (2024). To make a fair comparison with LLaVA-1.5-7B equivalent structure, we train an MLP projector and finetune pretrained LLM with LoRA. The default training setting uses Adam optimizer with minibatch size 4 and initial learning rate 1e-4. We use the linear schedule to decrease the learning rate linearly from the initial value.

Hyperparameter settings. For pretrained model clustering, the threshold for vision encoder and LLMs are set to $t_{ve} = 0.7$ and $t_{llm} = 0.8$, respectively. And the warmup round for the feature projector is 10. When performing an efficient evaluation, we set both $topk_{inter}$ and $topk_{intra}$ to 3, which means that the Top-3 clusters will be selected in inter-cluster evaluation and Top-3 candidates will be selected in intra-cluster evaluation with the early stopping mechanism. We use $\eta = 2$ as the default reduction factor, which is consistent with typical SHA settings. We further set $p = 3$ and $\delta = 5e - 5$ by default for scaling prediction. We will discuss the effect of hyperparameters in Appendix D.2.

Baseline. We compare Mordal with four model selection baselines: EMMS Meng et al. (2023), LogME You et al. (2021), LEEP Nguyen et al. (2020a), and NLEEP Li et al. (2021). For a fair comparison, we use the same alignment dataset and ensure that the total training time allocated to all candidates matches Mordal's overall GPU hour budget. Specifically, we divide Mordal's total training time by the number of candidates, and assign this budget to each candidate. Each VLM candidate is partially trained using this fixed alignment time. We then extract features from the trained model and compute label representations using three foundation models: CLIP Radford et al. (2021), BERT Devlin et al. (2019), and GPT-2 Radford et al. (2019). These features and label representations are used to compute the transferability metrics required by EMMS, LogME, LEEP, and NLEEP. This comparison setup follows the standard protocol used in EMMS Meng et al. (2023), ensuring consistency and fairness in evaluating model selection quality under limited compute.

Metric. To evaluate ranking accuracy, we compare candidates ranked by performance in grid search with those ranked by elimination order in Mordal. Ideally, if a candidate ranks higher in grid search, it should also rank higher in Mordal. This can be captured by Kendall's τ coefficient defined as:

$$\tau = \frac{2}{M(M-1)} \sum_{1 \le i < j \le M} sgn(T_i - T_j) sgn(S_i - S_j) \tag{4}$$

where M is the total number of candidates and $sgn()$ is the sign function. A perfect ranking match results in $\tau = 1$. To further focus on top-performing candidates, we adopt the weighted Kendall's coefficient τ_w, which is previously used in You et al. (2021); Vigna (2015). The details for implementing τ_w can be found in SciPy Virtanen et al. (2020).

D.1 Evaluation Time Breakdown

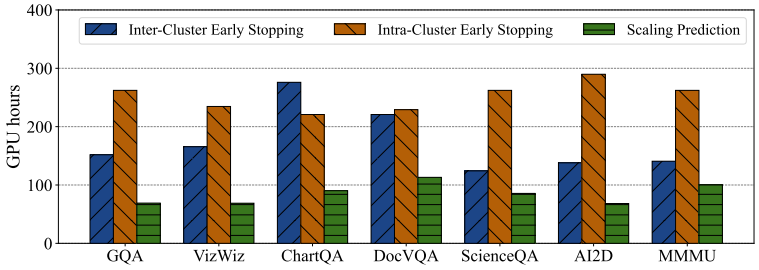


Figure 8: Total evaluation time breakdown for Mordal on seven datasets.

As shown in Table 2, When evaluating Mordal on different tasks, the total evaluation time required is different. To investigate the differences in time consumption, we analyze the breakdown time for each component of Mordal and present the result in 8. As expected, the early stopping stage consists of most of the evaluation time, and the prediction only takes a small part of time. This is because

the scaling prediction is only performed for the candidates left after early stopping. The time varies depending on when the model is converged and scaling is observed. The time for inter-cluster and intra-cluster early stopping depends on the number of clusters, controlled by t_{ve} and t_{llm}. The cluster is generally less obvious for difficult tasks (e.g., ChartQA and DocVQA), leading to many clusters with only one candidate. As fewer candidates are eliminated, the total evaluation time increases.

D.2 Sensitivity Analysis

Category	Config	Time	Top-1 Score	τ
Mordal	Default	483	66.4	0.81
Candidate	$t_{ve} = 0.5$	446	66.4	0.52
Clustering	$t_{ve} = 0.9$	1041	66.4	**0.86**
Inter-Cluster	$topk_{inter} = 2$	417	66.4	0.73
Evaluation	$topk_{inter} = 4$	564	66.4	0.83
Intra-Cluster	$topk_{intra} = 2$	**451**	**66.4**	0.81
Evaluation	$topk_{intra} = 4$	522	66.4	0.81
Prediction	$p = 4$	501	66.4	0.81

Table 6: Summary of sensitivity analysis for GQA.

Sensitivity analysis explores how key hyperparameters affect Mordal's performance and efficiency, focusing on clustering thresholds t_{ve} and t_{llm}, exploration parameters s_{inter} and s_{intra}, and scaling prediction p. Careful tuning of these parameters ensures efficient operation without compromising Mordal's ability to select top-performing models. Generally, Mordal is robust and consistently identifies the best-performing model across most hyperparameter settings.

Effect of clustering hyperparameters. The clustering threshold t_{ve} and t_{llm} significantly affects Mordal's performance and efficiency. As shown in Table 6, a smaller threshold $t_{ve} = 0.5$ creates fewer, larger clusters based on the LLM's general characteristics, which will lower the τ value by missing finer distinctions and discarding strong candidates with other LLM backend. On the other hand, a larger threshold $t_{ve} = 0.9$ results in more, smaller clusters, capturing subtle differences and improving the τ value but increasing the number of candidates to evaluate during scaling prediction, leading to longer searching time. Balancing the threshold is crucial to ensure diverse clusters while keeping the computational cost reasonable.

Effect of inter- and intra-cluster hyperparameters. Based on Table 6, inter-cluster exploration generally has a greater impact on Mordal's performance than intra-cluster exploration. A smaller $topk_{inter}$ reduces the number of clusters evaluated, speeding up the search but lowering the τ value by excluding promising clusters aggressively. A larger $topk_{inter}$ explores more clusters, increasing search time but improving the τ value by retaining diverse clusters. Intra-cluster early stopping affects candidate selection within clusters, with smaller $topk_{intra}$ focusing on fewer candidates and larger $topk_{intra}$ exploring more candidates for scaling prediction. However, its influence is smaller, as clusters already limit diversity. Properly balancing these $topk_{inter}$ and $topk_{intra}$ ensures efficient exploration and strong performance.

Effect of scaling prediction hyperparameters. The scaling prediction parameter p has minimal impact on Mordal's overall performance but affects search time. Increasing p beyond appropriate values adds to the computational cost without improving results. In practice, $p = 3$ is sufficient for constructing the linear regression model used in scaling prediction.

D.3 Impact of Vision Encoders vs. LLMs

To understand the respective contributions of vision encoders and LLMs, we evaluated 4×4 combinations of LLMs and vision encoders on two tasks: AI2D and VizWiz. Results are shown in Tables 9a and 9b. On AI2D, the choice of LLM dominates the performance: differences across LLMs (e.g.,

Model	CLIP	SigLIP	DFN	InternViT
Vicuna	54.8	53.0	52.7	53.4
Mistral	50.9	50.5	51.8	52.1
Qwen	61.2	65.2	64.1	61.8
LLaMA3	58.2	60.1	58.8	57.0

(a) AI2D

Model	CLIP	SigLIP	DFN	InternViT
Vicuna	41.2	44.8	34.4	30.7
Mistral	45.1	46.9	35.6	31.3
Qwen	43.3	44.6	38.9	33.4
LLaMA3	37.9	38.1	35.2	32.5

(b) VizWiz

Figure 9: Performance across combinations of LLMs and vision encoders on AI2D and VizWiz.

Qwen vs. Mistral) are significantly larger than differences across vision encoders within each row. On VizWiz, vision encoders dominate—replacing InternViT with SigLIP improves accuracy by 6–15% across all LLMs. These findings are consistent with prior work (Chen et al., 2024a; Wang et al., 2024), which shows that some tasks can be solved without a strong vision encoder.

Mordal accounts for these differences at a coarse granularity. For vision-centric tasks, the clustering stage effectively eliminates underperforming clusters associated with weak vision encoders. Among the remaining candidates—differing mainly in their language backbones—Mordal applies scaling prediction to estimate and rank their performance.

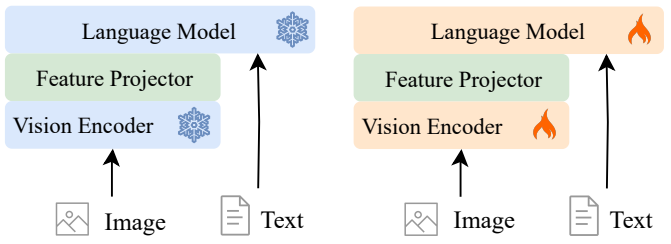
D.4 Small-scale Models

Dataset	Top-1 (Grid)	Top-1 (Mordal)	Time (Mordal)	Kendall's τ
GQA	62.4 (SigLIP-LLaMA)	62.4	189 (7.7×)	0.707
ChartQA	12.1 (SigLIP-Gemma)	12.1	217 (6.7×)	0.621
ScienceQA	66.8 (SigLIP-LLaMA)	66.8	166 (8.8×)	0.740

Table 7: Small-scale results. Mordal matches grid search in identifying the top-1 model while reducing training costs.

To evaluate scale robustness, we conducted additional experiments using the same seven vision encoders in the paper and smaller LLMs (i.e., Gemma-2-2B, LLaMA-3.2-3B, and Phi-3.5-mini-3B). The results are shown in Table 7. Mordal consistently selects the same top-1 model as full grid search across all tasks, while reducing total training costs by 6.7× to 8.8×. However, we observe that Kendall's τ is lower in the small-model setting. This is likely because smaller models exhibit more inconsistent behavior, making scaling prediction less stable (Zohar et al., 2025). In contrast, larger models tend to demonstrate more predictable scaling behavior, which makes Mordal more effective.

E Discussion



(a) Frozen pretrained modules (b) Updatable pretrained modules

Figure 10: Alternative alignment approaches for VLM instruction tunning.

Limitations. Mordal demonstrates promising results in efficiently selecting pretrained models fo VLM with small vision encoders and 7B LLMs under a single-request. However, certain limitations may be addressed to extend its utility and effectiveness. First, while Mordal significantly reduces search time, it does not guarantee perfect top-1 prediction. On challenging tasks, poor clustering or

early stopping may eliminate strong candidates. Second, Mordal's performance can be sensitive to both dataset and model scale; for instance, scaling prediction becomes less accurate with smaller models, which tend to exhibit higher variance and less predictable behavior. Despite these limitations, Mordal achieves a balance between efficiency and accuracy. It consistently identifies top-performing models with a speedup of up to $11.6\times$ over grid search, making it a practical solution for real-world VLM selection where exhaustive search is often infeasible.

VLM alignment. One must go through an *alignment* process to ensure that the individual components of the VLM are well integrated before using it. Developers integrate pretrained LLMs and visual encoders, training the projector from scratch using visual alignment datasets like VQA Antol et al. (2015) . During the alignment process, pretrained components may remain frozen or be further finetuned during alignment training Liu et al. (2023a). Recently, some proprietary models have employed end-to-end training without using any pretrained models Bai et al. (2023), but it is not common due to the excessive training cost. While Mordal addresses pretrained model selection in a popular VLM setting, it would be interesting to investigate its effectiveness under other VLM structures with and without pretrained components.

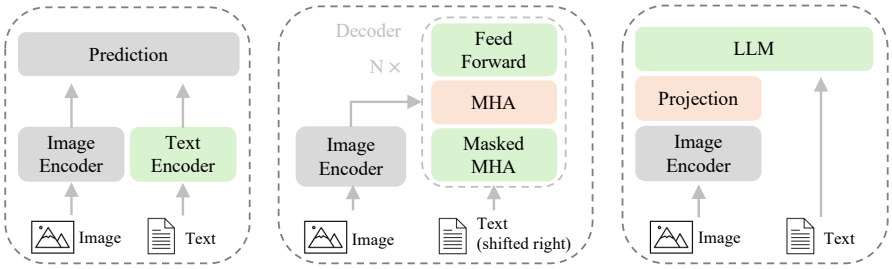
Non-MLP projectors. We acknowledge the existence of non-MLP projector architectures, such as the Q-former used in BLIP-2 (Li et al., 2023). However, Q-former is being replaced in recent VLMs. For example, BLIP-3 (Xue et al., 2024) discards Q-former in favor of a more efficient sampler mechanism. Currently, MLP-based projectors are the most prevalent choice in the VLM community due to their simplicity, effectiveness, and compatibility with diverse architectures. Furthermore, MLP projectors have been shown to generalize well beyond vision-language settings—for instance, in multimodal projection for text, audio, and video in NExT-GPT (Wu et al., 2024). While our current experiments focus on the MLP projector, Mordal is agnostic to the choice of projector architecture, as long as the projector outputs a fixed embedding. We will expand our discussion in the future version to clarify this point and outline how Mordal could be extended to handle non-MLP and multimodal projectors.

Extend to smaller and larger models. Mordal's design and evaluation have focused on small size (i.e., 7B) pretrained models, making it efficient and practical for scenarios with limited computational resources. However, extending Mordal to handle smaller (e.g., 1B) and larger models (e.g., 70B) introduces new challenges. For example, the computational overhead associated with larger models significantly increases, requiring more memory and longer processing times for alignment and evaluation. Mordal's current optimization strategies may not scale effectively under these conditions, necessitating further refinement to manage resource demands. Additionally, as shown in Appendix D.4, the similarity measurements and observational scaling law used to speed up evaluation could become less effective with smaller or larger models due to shifts in their feature spaces, potentially reducing the accuracy of candidate selection. To address these challenges, future iterations of Mordal must incorporate distributed computing frameworks and advanced resource allocation techniques. Adjustments to representation similarity metrics and scaling law formulations will also be essential to maintain the framework's robustness as models vary in size and complexity.

Similar requests among users. Mordal's current implementation is designed to optimize pretrained model selection for a single request at a time, which limits its efficiency in handling multiple similar tasks submitted by users. This approach overlooks opportunities to reduce redundant computations when user requests share overlapping requirements, leading to inefficient use of computational resources. By evaluating a shared set of model candidates for grouped tasks, Mordal could eliminate redundant computations and improve throughput. For example, implementing a caching mechanism to store and reuse results for previously evaluated models and tasks could further enhance resource efficiency. Addressing these limitations would enable Mordal to support multi-user environments and dynamic workloads more effectively.

F Related Work

Model selection. Training-free model selection methods such as EMMS Meng et al. (2023), LogME You et al. (2021), LEEP Nguyen et al. (2020a), and NLEEP Li et al. (2021) assess the transferability of pretrained features without requiring additional finetuning. However, these methods are primarily designed for architectures like CLIP and VisionEncoderDecoder models (as illustrated in



(a) Clip-based VLM (b) VisionEncoderDecoder VLM (c) LLM-based VLM

Figure 11: Vision-language models: (a) CLIP-based VLM aligns image and text embeddings via contrastive learning; (b) VisionEncoderDecoder VLM uses an encoder-decoder structure for classification and OCR tasks; (c) LLM-based VLM combines a vision encoder with a language model for multimodal interactions.

Figure 11a and Figure 11b). Other approaches assume that all candidates share the same architecture and differ only in pretraining datasets Tran et al. (2019). LLM-based selection methods Lin et al. (2024) are also not directly applicable to VLMs, as vision and language components are pretrained separately and not jointly aligned. Moreover, most existing techniques are tailored for classification or regression tasks, and struggle with the open-ended nature of multimodal generation. Given the growing number of open-source pretrained models and the unique challenges posed by multimodal alignment, there is a clear need for pretrained model selection approaches specifically designed for LLM-based VLMs (i.e., Figure 11c). Mordal addresses this gap by enabling efficient and alignment-aware selection of pretrained model combinations.

G LLM Usage

We used large language models solely to assist in polishing the writing of this paper. No part of the research ideation, experimental design, or analysis relied on LLMs.