Do Multimodal Foundation Models Truly Generalize? Exposing Failure Modes Across Perception, Reasoning, and Action

Supplementary Material

A. Metrics

For discrete action prediction in game environments (Overcooked), near-misses receive no partial credit in discrete control. We employ Exact Match Rate (EMR) as the primary metric, measuring the percentage of predictions that exactly match ground truth actions:

$$\text{EMR} = \frac{\text{Number of correct predictions}}{N} \tag{1}$$

High EMR indicates the model accurately predicts the correct action, while low EMR suggests the model frequently selects incorrect actions regardless of how semantically similar they might be to the ground truth.

We supplement EMR with micro-averaged precision, recall, and F1 scores, which aggregate metrics treating each prediction equally and reflecting overall prediction accuracy weighted by action frequency:

$$P_{\text{micro}} = \frac{\sum_c \text{TP}_c}{\sum_c (\text{TP}_c + \text{FP}_c)},\ R_{\text{micro}} = \frac{\sum_c \text{TP}_c}{\sum_c (\text{TP}_c + \text{FN}_c)},\ F1_{\text{micro}} = \frac{2P_{\text{micro}} R_{\text{micro}}}{P_{\text{micro}} + R_{\text{micro}}} \tag{2}$$

For multi-class single-label prediction tasks where action classes are mutually exclusive, an incorrectly predicted action is both a false positive for its predicted class and a false negative for the ground truth class. Micro metrics aggregate true positives, false positives, and false negatives globally across all timesteps before computing precision, recall, and F1. High micro precision indicates the model gets a large proportion of predictions correct overall, while low micro precision suggests frequent mismatches with ground truth. Micro metrics favor majority classes due to frequency weighting.

Macro-averaged precision, recall, and F1 compute per-class metrics first, then average across all classes, weighting each action class equally regardless of frequency:

$$P_{\text{macro}} = \frac{1}{C}\sum_{c=1}^{C} \frac{TP_c}{TP_c + FP_c},\ R_{\text{macro}} = \frac{1}{C}\sum_{c=1}^{C} \frac{TP_c}{TP_c + FN_c},\ F1_{\text{macro}} = \frac{1}{C}\sum_{c=1}^{C} \frac{2P_c R_c}{P_c + R_c} \tag{3}$$

This reveals systematic biases toward specific actions or failure on rare but critical actions. We primarily report macro recall, which avoids both the majority-class bias of micro metrics and the rare-class sensitivity of macro precision or F1 [1]. Macro metrics are particularly valuable for detecting whether models handle rare but strategically important actions (e.g., specialized game actions, object interactions) as reliably as common actions (e.g., basic movement). Low macro recall combined with high micro recall indicates the model performs well on frequent actions but poorly on rare ones. Large discrepancies between micro and macro metrics indicate aggregate performance masking poor rare-action understanding.

We additionally report invalid predictions (outputs outside the valid action space) and calculate precision excluding these, separating formatting from semantic errors [1]. Invalid predictions occur when models generate outputs that do not correspond to any valid action in the action space. For VLMs producing probability distributions, outputs are invalid if they fail to parse into the expected format (e.g., malformed JSON, probabilities not summing to 1). A high invalid percentage indicates the model struggles to constrain outputs to the target action space, representing a distinct failure mode from semantic misclassification. Precision calculated excluding invalid predictions reveals model performance when outputs are well-formed, isolating semantic understanding from formatting issues. For Overcooked's multi-agent coordination, we decompose the joint action space into per-player accuracies, diagnosing whether coordination failures stem from individual behavior errors or misunderstanding inter-agent dependencies. High individual player accuracy combined with low joint accuracy suggests the model understands each agent's behavior independently but fails to capture coordination patterns.

For continuous action prediction in robotics tasks from OpenX, raw error values are incomparable across datasets due to heterogeneous action spaces with different scales. We employ Mean Absolute Error (MAE) and Mean Squared Error (MSE) as base metrics, computing per-timestep distance averaged across action dimensions:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} |\hat{\mathbf{a}}_i - \mathbf{a}_i|, \quad \text{MSE} = \frac{1}{N}\sum_{i=1}^{N} \|\hat{\mathbf{a}}_i - \mathbf{a}_i\|^2 \tag{4}$$

MAE provides interpretable error magnitude in the action space's native units, while MSE penalizes large deviations more heavily by squaring errors, making it sensitive to outliers. Low MAE indicates predicted actions are close to ground truth on average, while high MAE suggests significant deviations. However, raw MAE and MSE values are not comparable across datasets with different action space scales (e.g., joint angles in radians vs. gripper forces in Newtons).

We therefore normalize model errors relative to a baseline predictor that always outputs the training set mean action for each dimension, yielding Approximate Relative MAE (Approx RelMAE) as our primary cross-dataset comparison metric:

$$\text{Approx RelMAE} = \frac{\frac{1}{N}\sum_{j=1}^{N} \text{MAE}_{\text{model},j}}{\frac{1}{N}\sum_{j=1}^{N} \text{MAE}_{\text{baseline},j}} = \frac{\sum_{j=1}^{N}\sum_{i=1}^{D} |\hat{y}_{j,i} - y_{j,i}|}{\sum_{j=1}^{N}\sum_{i=1}^{D} |\bar{y}_{\text{train},i} - y_{j,i}|} \quad (5)$$

where $\hat{y}_{j,i}$ is the model's predicted action for dimension i at timestep j, $y_{j,i}$ is the ground truth action, $\bar{y}_{\text{train},i}$ is the mean action value for dimension i computed over the training set, N is the number of timesteps, and D is the action dimensionality. The baseline predictor represents an uninformed strategy that always predicts the central tendency of the training distribution regardless of visual input or context. Approx RelMAE values less than 1.0 indicate the model outperforms this naive baseline, while values greater than 1.0 indicate worse-than-baseline performance. This normalization enables cross-dataset comparison by measuring model performance relative to dataset-specific baselines rather than using absolute error scales. For invalid predictions (NaN, inf, or malformed outputs), we assign the baseline error for that timestep, ensuring invalid outputs receive appropriate penalties without arbitrarily inflating metrics.

We analogously define Approx RelMSE by replacing absolute errors with squared errors:

$$\text{Approx RelMSE} = \frac{\sum_{j=1}^{N} \|\hat{\mathbf{y}}_j - \mathbf{y}_j\|^2}{\sum_{j=1}^{N} \|\bar{\mathbf{y}}_{\text{train}} - \mathbf{y}_j\|^2} \quad (6)$$

Values below 1.0 indicate the model outperforms the naive baseline in squared error; values above 1.0 indicate worse-than-baseline performance.

Continuous control errors exhibit fat-tailed distributions with occasional catastrophic predictions, so we additionally report quantile-filtered Approx RelMAE (excluding errors beyond the 5th and 95th percentiles), maximum relative MAE, and the proportion of predictions exceeding 3x the median error threshold. These outlier-specific metrics capture failure modes that mean-based metrics obscure. Quantile-filtered Approx RelMAE reveals typical performance when extreme outliers are excluded. Maximum relative MAE quantifies how much the worst-case error deviates from the median error; values significantly greater than 1 indicate some extremely poor predictions that skew mean metrics. The proportion of predictions exceeding 3x median threshold identifies the frequency of severe failures.

For vision-language tasks, we employ task-specific metrics based on output format. Exact Match Rate (EMR) measures the percentage of successfully parsed predictions that exactly match ground truth answers for tasks with discrete answer sets. Tasks span diverse formats: multiple-choice selection (PIQA), free-form text responses (SQA3D, RoboVQA), structured function calls (BFCL), and object classification. For ODINW, we reformulate the object detection task as a classification problem where models predict object category presence rather than bounding box coordinates, enabling unified evaluation across VLMs and VLAs. We parse outputs into expected formats, flagging parsing failures or constraint violations as invalid. High EMR indicates strong task understanding and correct output generation, while low EMR suggests either semantic misunderstanding or formatting errors. We distinguish between these failure modes by separately tracking invalid predictions.

For free-form vision-language tasks where multiple valid textual responses exist, we measure semantic similarity using cosine similarity between sentence embeddings of predicted and reference answers:

$$\text{Sim}(\hat{y}, y) = \frac{\text{emb}(\hat{y}) \cdot \text{emb}(y)}{\|\text{emb}(\hat{y})\|\|\text{emb}(y)\|} \quad (7)$$

Embeddings are computed using all-MiniLM-L6-v2 [2]. This metric captures semantic similarity even when predicted answers use different wording than reference answers for tasks like SQA3D and RoboVQA. Similarity scores near 1.0 indicate high semantic overlap, while scores near 0 indicate unrelated or contradictory answers. This provides a more nuanced evaluation than strict string matching for tasks where multiple phrasings are valid. We report invalid prediction rates separately, as formatting failures represent a distinct failure mode from semantic errors. Models may demonstrate correct reasoning but fail to produce outputs in the required structure, or conversely, may generate well-formed but semantically incorrect outputs.

We employ diagnostic tools to isolate failure modes. For discrete action tasks, we report clipped metrics where invalid predictions are clipped to valid action space bounds, establishing performance upper bounds via output normalization. Large gaps between clipped and unclipped metrics indicate that output formatting issues, rather than semantic understanding, limit performance. For probabilistic predictions in multi-agent tasks like Overcooked, we quantify calibration using Brier MAE, which measures mean absolute error between predicted probability distributions and one-hot ground truth:

$$\text{Brier MAE} = \frac{1}{N}\sum_{t=1}^{N}\sum_{i=1}^{R} |p_{ti} - y_{ti}| \quad (8)$$

where p_{ti} is the predicted probability for class i at timestep t, y_{ti} is the one-hot ground truth, and R is the number of classes. Following [1], we use MAE rather than MSE to maintain interpretability while reducing outlier sensitivity. This metric penalizes confident but incorrect predictions more severely than uncertain errors, assessing whether models assign high probability to likely actions and low probability to unlikely alternatives based on context. Good Brier scores combined with poor EMR indicate well-calibrated uncertainty but indecisive predictions, while poor Brier scores with good EMR suggest overconfident but occasionally correct predictions. Brier MAE has a maximum

value of 2, achieved when the model assigns probability 1 to an incorrect action.

Listing 1 is an example multi-turn conversation that's part of the prompt for the BFCL benchmark.

Listing 1. Example multi-turn BFCL prompt used in our evaluation.

```
  "conversation": [
    {
      "role": "system",
      "content": "You are an AI assistant
           that can call functions to
           complete tasks. You will be
           presented with conversation
           histories where each turn may
           require function calls.

For each turn, analyze the conversation
     history, which may include previous
     assistant responses in addition to user
      prompts, and respond with the correct
     function to call.
Format each function call as: function_name(
     param1=value1, param2=value2, ...)
Use only the exact function names available
     in the provided set of functions and
     append appropriate parameters.
Output only the function calls, no
     explanations or additional text."
    },
    {
      "role": "user",
      "content": "Initial Environment
           Configuration:
{
  "conversation_id": "multi_turn_001",
  "turns": [
    [
      "I need to book a flight from San
           Francisco to New York for next
           Monday."
    ],
    [
      "Great! Now book a hotel near JFK
           airport for the same dates."
    ],
    [
      "Can you also rent a car for the
           duration of the trip?"
    ]
  ],
  "ground_truth_functions": [
    [
      {
        "function": "book_flight",
        "arguments": {
          "origin": "SFO",
          "destination": "JFK",
          "date": "2024-03-18"
        }
      }
    ],
    [
      {
        "function": "book_hotel",
        "arguments": {
          "location": "near JFK airport",
          "check_in": "2024-03-18",
          "check_out": "2024-03-22"
        }
      }
    ],
    [
      {
        "function": "rent_car",
        "arguments": {
          "pickup_location": "JFK airport",
          "pickup_date": "2024-03-18",
          "return_date": "2024-03-22"
        }
      }
    ]
  ],
  "initial_config": {
    "available_apis": ["booking_service", "
         travel_assistant"]
  },
  "num_turns": 3
}
```

References

[1] Pranav Guruprasad, Yangyue Wang, Sudipta Chowdhury, Harshvardhan Sikka, and Paul Pu Liang. Benchmarking vision, language, & action models in procedurally generated, open ended action environments, 2025.

[2] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

Table 1. Prompt templates and expected outputs used in our benchmark. Example outputs are representative of correctly formatted responses. Prompt templates are slightly different among models.

Dataset	Prompt Template	Ex. Input	Out. Format	Ex. Output	Metric
PIQA	You are evaluating physical commonsense. Given two choices, select the physically plausible one. Output only the index of the correct solution, and nothing else...⟨*Question*⟩	*Question:* "goal": "How to make a simple ice pack at home", "sol1": "Take a clean sock and fill it with uncooked rice. Tie the end and place it in the freezer for a few hours.", "sol2": "Take a clean sock and fill it with cooked pasta. Tie the end and place it in the freezer for a few hours.",	Binary choice (`0` or `1`)	`0`	EMR
SQA3D	You are a vision-language model specializing in 3D scene understanding and question answering... Instructions: Respond with only your answer. Do not provide explanations or reasoning ⟨*Scene*⟩⟨*Question*⟩	"situation": "I am standing by the ottoman on my right facing a couple of toolboxes.", "alternative_situation": "I just placed two backpacks on the ottoman on my right side before I went to play the piano in front of me to the right.", "question": "What instrument in front of me is ebony and ivory?", "image": "⟨scene image as numpy array with shape (H, W, 3)⟩"	Short natural-language phrase	`"piano"`	EMR & Semantic Similarity
RoboVQA	You are a Visual-Language Model Assistant that specializes in answering questions about robotics tasks based on given images representing the robot's environment... Output formatting rules:... ⟨*Image*⟩⟨*Question*⟩	"question": "Task and Context: Navigate to the kitchen counter and pick up the red mug. Question: What should I do next?", "image": "⟨robot's camera view as numpy array with shape (H, W, 3)⟩"	Yes/No/ Action phrase	`"grasp the red mug handle"`	EMR & Semantic Similarity
ODinW	You are a specialized Visual-Language Model Assistant that identifies the object in a given image and selects the best option possible from the options provided... The answer must be a single integer... ⟨*Question*⟩	*Question:* "image": "⟨cropped object image as numpy array with shape (H, W, 3)⟩", "question": "What object is shown in this image from the BCCD dataset? Option 0: RBC Option 1: WBC Option 2: Platelets Output the number (0-2) of the correct option only.", "category_name": "RBC", "options": ["RBC", "WBC", "Platelets"],	Single integer label (`0, 1, 2...`)	`"0"`	EMR & F1
BFCL	You are an AI assistant that can call functions to complete tasks. You will be presented with conversation histories where each turn may require function calls... Output only the function calls, no explanations or additional text. ⟨*Conversation*⟩	**check the Supplementary Material for an example conversation Listing1**	List of structured function calls	`[mkdir("data"), move("a.txt", "data/")]`	EMR & Semantic Similarity
Overcooked (Action)	Layout ⟨*Layout*⟩. Time: ⟨*time*⟩ elapsed... Actions: 0-5: Player 0: North... Output joint action index as single value between 0-35.	*Observation:* "layout_name": "cramped_room", "image_observation": "⟨game screenshot as numpy array with shape (H, W, 3)⟩", "time_elapsed": 19.5,	Single integer label (`0, 1...`)	`0`	EMR & F1
Overcooked (Probs)	We are running a simulation for two AI agents cooperatively playing Overcooked in layout ⟨*Layout*⟩, a kitchen coordination game. ⟨*Observation*⟩	-	List of 36 probs	`[0.12, 0.04, ..., 0.03]`	Brier MAE
Open-X Embodiment	You are an AI agent performing the task ⟨*Input*⟩. Predict the next action in the control sequence.	"text_observation": "pick up the red block and place it in the bin", "image_observation": "⟨robot camera view as numpy array with shape (H, W, 3)⟩"	Continuous 7D or 8D action vector	`[0.12, -0.05, 0.08, 0.0, 0.0, 0.3, 1.0]`	RelMAE & RelMSE
Open-X Quadrupedal	You are an AI agent to solve the task... The actions available: ⟨*Action Schema with Min/Max/Mean Stats*⟩... You must generate your output keeping the following format: A list starting with '['..."	-	Continuous 12D action vector	`[0.15, -0.07, 0.02, ..., 1.0]`	RelMAE & RelMSE