FPBENCH: A Comprehensive Benchmark of Multimodal Large Language Models for Fingerprint Analysis

Ekta Gavas
New York University
eg4131@nyu.edu

Sudipta Banerjee
University of Wyoming
sbanerj3@uwyo.edu

Chinmay Hegde
New York University
chinmay.h@nyu.edu

Nasir Memon
New York University
memon@nyu.edu

A. Statistics of evaluation question prompts

The cumulative statistics for the questions in FPBENCH evaluation is provided in Tab 1.

Table 1. Key statistics of questions in FPBENCH.

Statistic	Number
Total questions	4940
Total categories	5
Total tasks	8
Public datasets used	6
Questions with multiple images	3220 (65%)
Questions with single image	1620 (33%)
Questions with only text	100 (2%)
Total images in all questions	13654
Unique number of images	9034
Unique question templates	419
Maximum question length	1275
Maximum option length	160
Average question length	71.59
Average option length	6.80
Total options in each question	3 or 4
Frequency of A as correct option	1330 (26.92%)
Frequency of B as correct option	1261 (25.52%)
Frequency of C as correct option	1299 (26.29%)
Frequency of D as correct option	1050 (21.69%)

B. Dataset statistics

Tab 2 represents the fingerprint dataset statistics used in FPBENCH. Note that the datasets were cleaned to remove any latent prints, palm prints, digital (RGB) fingerprint photos or 3D prints prior to using them in FPBENCH.

C. Example ACE-V sheet

Fig 1 shows an example ACE-V sheet for fingerprint analysis and evaluation for a pair of prints. The sheet concludes "Individualization" as Level 1 and Level 2 features are in agreement.

Table 2. Statistics of fingerprint datasets used for FPBENCH

Dataset	No. of Fingers	No. of Impressions	Total Images
FVC2000 [7]	440	8	3520
FVC2002 [8]	440	8	3520
FVC2004 [9]	440	8	3520
NIST SD302d [5]	2000	1-3	5141
NIST SD301a [4]	240	1-15	4366
GenPrint [6]	10000	15	150000
Anguli [1]	10000	-	10000

Table 3. Configuration settings for closed-source models in FPBENCH

Model	zero-shot/chain-of-thought	
	Reasoning	maxOutputTokens
GPT-5 [12]	reasoning_effort='minimal'/'medium'	32 / 128,000
Gemini 2.5 Pro [3]	thinkingBudget=128/dynamic	136 / 65,536

D. Proprietary models API config

Tab 3 refers to API configuration for proprietary models GPT-5 and Gemini 2.5 Pro in zero-shot and CoT settings.

E. Additional Results

The change in performance under different evaluation settings is shown in Tab 4. Fig 2 depicts the performance of all the models across all the tasks in the form of a heatmap. This helps in understanding the individual and average performance of the models across all the tasks. The radar plot in Fig 3 indicates the accuracy with increasing value as one moves away from the center (0%) towards the outer periphery (100%); each axis on the concentric circle corresponds to a single task. This depicts that the model with a larger area

ID #	N/A
Analysis must be consistent with SWGFAST terminology and definitions, and shall include, but not be limited to, the following elements:	

ANALYSIS

Name: Image 1	DOB: N/A	Case ID: N/A	Name: Image 2	DOB: N/A	Case ID: N/A
Level 1: Finger: Distal Phalange, Whorl Level 2: Sufficient			Level 1: Finger: Distal Phalange, Whorl Level 2: Sufficient		
Level 3 Visible? NO			Level 3 Visible? NO		
QUALITY	Medium High		QUALITY	Medium High	
SUITABLE FOR COMPARISON	☒		SUITABLE FOR COMPARISON	☒	
Fingerprint ☒		Palm print ☐			Footprint ☐

COMPARISON

LEVEL 1 AGREEMENT ☒	LEVEL 2 AGREEMENT ☒	LEVEL 3 AGREEMENT ☐

EVALUATION

INDIVIDUALIZATION ☒	EXCLUSION ☐	INCONCLUSIVE ☐

Figure 1. Example ACE-V sheet from the ACE-V analysis task on a sample fingerprint pair in FPBENCH. The template is referenced from Document 9 of SWGFAST documentations [11].

(in all directions) has an overall better performance across all categories and tasks.

F. Zero-shot vs. CoT output comparison

Manual review of the CoT response of Qwen3-VL-32b model revealed reasoning inconsistencies (Fig 4). Despite concluding a match, the model assigned incorrect and dissimilar pattern classes to the images, indicating high-level understanding without fine-grained discrimination. This opens the path for domain adaptation that can potentially improve the CoT reasoning capabilities of the models.

G. Tool Retrieval Questions Generation:

Following the approach in [10], we designed a detailed prompt with 10 APIs and a total of 22 functions to generate the FPBENCH questions. We included APIs covering a wide range of fingerprint understanding tasks, complying with real-world scenarios and systems. We generated a total of 150 questions each from GPT-5 and Gemini 2.5 Pro, out of which 100 were selected. We manually reviewed and rephrased the questions to maintain diversity and correctness. The detailed prompt for question generation is provided below.

Prompt for Generating Tool Retrieval Questions:

You are an AI tasked with generating complex, real-world scenario questions to assess a model's ability to select the correct API and function calls to accomplish nuanced tasks. Use the list of APIs and functions provided below.

1. **Pattern Classification:**
   api_name: `api_1`
   - `classify_pattern`
     Description: Predicts the pattern class in a given fingerprint image.
     Input: `np.ndarray` or `str` - The input fingerprint image.
     Output: `str` - The predicted pattern ('loop', 'whorl', or 'arch').
   - `get_pattern_probabilities`
     Description: Returns probabilities for each pattern class.
     Input: `np.ndarray` - The input fingerprint image.
     Output: `dict` - Probabilities for each pattern class.
   - `match_pattern`
     Description: Checks whether the fingerprint pair has the same pattern class.
     Input: Two `str` - Two fingerprint pattern classes.
     Output: `bool` - True if fingerprint patterns match, False otherwise.
2. **Matching:**
   api_name: `api_2`
   - `extract_features`
     Description: Extracts fingerprint features from fingerprint images.
     Input: `np.ndarray` or `str` - The input fingerprint image.
     Output: `np.ndarray` - Feature vector for the fingerprint.
   - `compare_features`
     Description: Compares two fingerprint feature vectors for a match.
     Input: Two `np.ndarray` - The feature vectors of

Table 4. Change in performance (accuracy %) under chain-of-thought compared to zero-shot evaluation setting in FPBENCH. Negative values suggest a decrease in performance in the chain-of-thought as compared to the zero-shot setting.

	Overall (5000)	Pattern (800)	Minutiae (800)	Orientation (400)	Verification (800)	Sensor (800)	Real/Synthetic (600)	ACE-V Analysis (600)	Tools Retrieval (100)
Open source MLLMs (<4B parameters)									
LLaVA-OneVision-0.5b-OV	-18.99	-3.33	3.44	-3.2	-1.0	-5.97	-1.76	-1.17	-6.0
Qwen3-VL-2b-Instruct	3.23	-4.56	1.85	2.21	1.74	9.33	-3.16	4.32	-8.5
Open source MLLMs (4B - 13B parameters)									
Gemma3-4b	-33.32	-12.07	-4.68	-3.94	6.09	-15.42	-2.63	2.33	-3.0
Chameleon-7b	-15.07	0.99	0.12	-2.71	-2.98	-1.24	-8.6	-2.65	2.0
LLaVA-v1.5-7b	-31.95	-13.3	-0.25	-8.62	-3.73	-3.98	-1.23	0.16	-1.0
LLaVA-NeXT-Interleave-7b	1.28	-1.23	-1.11	-3.2	0.12	-0.38	-1.75	-0.17	9.0
LLaVA-OneVision-7b-SI	-39.02	-11.57	-5.55	-5.67	-3.48	-1.12	-2.63	-1.0	-8.0
LLaVA-OneVision-7b-OV	-13.64	-2.96	5.17	-2.95	1.24	0.99	-2.63	-0.5	-12.0
DeepSeek-VL-7b	-11.06	1.97	1.23	-3.2	-0.25	-0.12	-0.35	-0.34	-10.0
Qwen3-VL-8b-Instruct	-21.89	-7.63	-7.02	-1.97	-6.10	-4.35	-8.77	13.95	0.00
Monkey-Chat	-3.17	-1.11	-1.10	-1.97	-1.37	-1.24	-0.88	0.50	4.00
Idefics2-8b	-5.00	-5.91	2.46	0.74	-0.74	0.62	-0.35	0.17	-2.0
InternVL3-8b	-62.16	-5.55	-3.7	-0.98	-6.84	-14.8	-13.16	-9.13	-8.0
Idefics-9b-Instruct	-9.19	-3.45	-0.74	-0.98	3.11	-2.98	0.18	-0.33	-4.0
Gemma3-12b	-55.78	-11.33	-4.31	-5.66	-4.48	-11.57	-5.44	-1.99	-11.00
Open source MLLMs (>13B parameters)									
LLaVA-v1.5-13b	-34.09	-9.61	2.09	-7.63	-1.24	-6.1	1.4	0.0	-13.0
Qwen3-VL-32b-Instruct	-41.26	-15.02	-6.15	-9.36	-2.61	-6.72	-1.23	-0.17	0.0
InternVL3-38b	-52.98	-4.19	-0.62	-3.94	-6.47	-20.9	-7.54	-4.32	-5.0
Proprietary MLLMs									
GPT-5	34.03	-4.31	7.15	-7.14	9.83	-6.47	4.74	30.23	0.0
Gemini 2.5 Pro	-100.76	-12.19	-19.46	-10.84	-17.53	-9.82	5.62	-35.54	-1.0

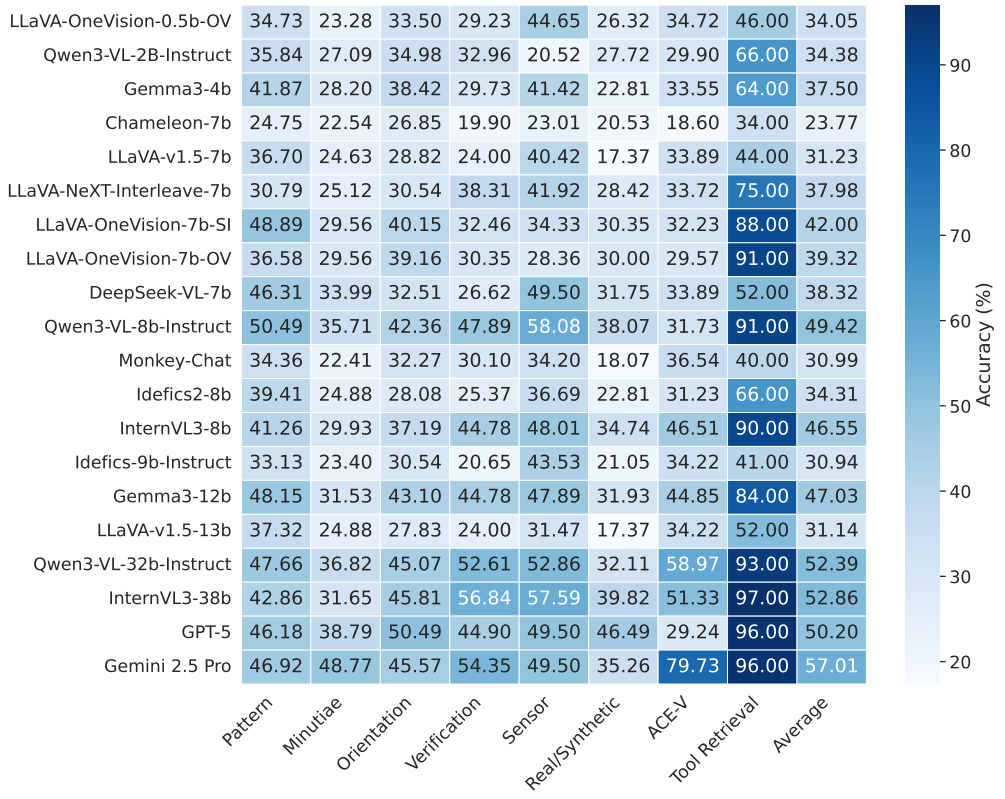

Figure 2. Accuracy (%) of all models across various fingerprint tasks presented in the form of a heatmap. The Tool Retrieval task appears to be the best-performing task across a majority of the models, whereas all the models struggle to distinguish between real and synthetic fingerprints on the Real vs. Synthetic classification task.

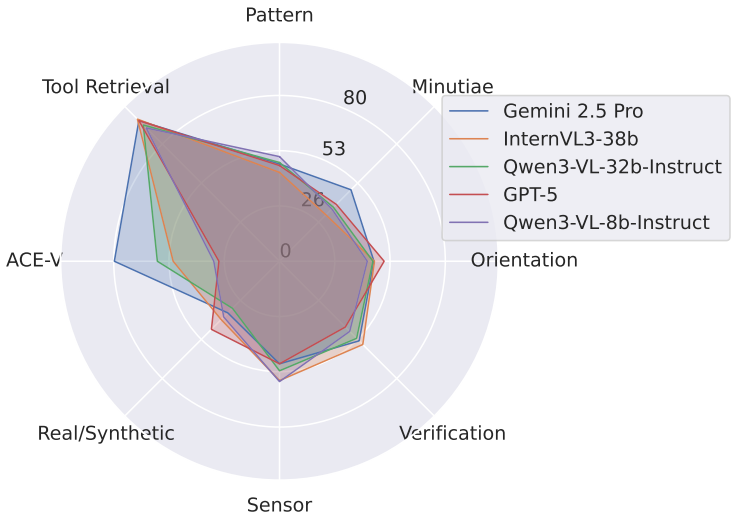

Figure 3. Performance (%) of top-5 best performing models across all tasks on zero-shot prompting.

Prompt:

Identify the matching fingerprint pair among the three images.
(A) All of them
(B) Image 2 and Image 3
(C) All are from different fingers
(D) Image 1 and Image 3

ZERO – SHOT RESPONSE:
B

CHAIN-OF-THOUGHT RESPONSE:
Let's analyze the three fingerprint images step by step:
-**Image 1**: This is a very low-resolution, grainy, and distorted fingerprint. The ridge patterns are broken and fragmented, making it difficult to discern the exact pattern type. However, it appears to have a complex, possibly whorl-like structure with multiple bifurcations and ridges curving in different directions.
-**Image 2**: This is a clearer fingerprint with a distinct **loop pattern**. The ridges enter from one side, curve around, and exit on the same side, forming a single delta and core. The pattern is well-defined and typical of a standard loop.
-**Image 3**: This fingerprint also shows a **loop pattern**, similar to Image 2. The ridges curve in a similar direction, and the overall structure is nearly identical in terms of ridge flow, core placement, and delta location.
Now, comparing Image 2 and Image 3:
- The ridge flow, direction, and overall pattern are nearly identical.
- The core and delta positions appear to be in the same relative locations.
- The quality and clarity are similar, suggesting they may be from the same finger, possibly taken at different times or under slightly different conditions.
Image 1, on the other hand, is of much lower quality and appears to have a different ridge structure, possibly a whorl or a complex loop, which does not match the clear loop patterns in Images 2 and 3.
Therefore, the most reasonable conclusion is that **Image 2 and Image 3** are the matching pair.
Final Answer: B

Figure 4. Comparison between responses produced by Qwen3-VL-32b[2] on the 'verification' task from zero-shot and CoT prompting.

two fingerprint images.
Output: `bool` - True if fingerprints match, False otherwise.

- `get_matching_score`
  Description: Get matching score for a given fingerprint image pair.
  Input: Two `str` or `np.ndarray` - Input fingerprint image pair.
  Output: `int` - Matching score.
- `identify_fingerprint`
  Description: Identifies a fingerprint impression by comparing a feature vector to a database.
  Input: `np.ndarray` features and `dict` database - Feature vector to identify and known features.
  Output: `str` or `None` - Label of the identified subject/impression/source, or `None` if no match.

3. **Minutiae:**
   api_name: `api_3`
   - `extract_minutiae`
     Description: Extracts fingerprint minutiae from a given fingerprint image.
     Input: `np.ndarray` or `str` - The input fingerprint image.
     Output: `np.ndarray` - List of x, y, theta, type of

minutiae points.
- `plot_minutiae_overlay`
  Description: Plots minutiae map over fingerprint image.
  Input: `str` or `np.ndarray` - Input fingerprint image.
  Output: `np.ndarray` - Overlaid minutiae over fingerprint as image.
- `get_minutiae_count`
  Description: Get count of total minutiae, ridge endings, and ridge bifurcations from minutiae points list.
  Input: `np.ndarray` - List of minutiae locations (x, y, theta, type).
  Output: Three `ints` - Number of minutiae, ridge endings, and ridge bifurcations.

4. **Orientation:**
   api_name: `api_4`
   - `get_orientation_angles`
     Description: Get orientation angles for every $k \times k$ block in input fingerprint image.
     Input: `str` or `np.ndarray` and `int` - Input fingerprint image and block size k.
     Output: `np.ndarray` - 2D array of orientation angles.
   - `plot_orientation_map`
     Description: Draw orientation flow map on fingerprint image.
     Input: `str` or `np.ndarray` and `np.ndarray` - Input fingerprint image and orientation angles.
     Output: `np.ndarray` - Output image of orientation flow map overlaid on fingerprint.
5. **Sensor Classification:**
   api_name: `api_5`
   - `predict_sensor_type`
     Description: Predict the type of sensor used to capture the fingerprint image.
     Input: `str` or `np.ndarray` - Input fingerprint image.
     Output: `str` - Predicted sensor type ('optical', 'thermal', 'capacitive', or 'none').
   - `get_sensor_probabilities`
     Description: Returns probability values for each sensor type.
     Input: `str` or `np.ndarray` - Input fingerprint image.
     Output: `dict` - Probabilities for each sensor type.
6. **Real vs Synthetic Classification:**
   api_name: `api_6`
   - `predict_real_synthetic`
     Description: Predict if the input fingerprint image is real or synthetic.
     Input: `str` or `np.ndarray` - Input fingerprint image.
     Output: `str` - Predicted class ('real' or 'synthetic').
   - `get_real_probabilities`
     Description: Returns probabilities for each class.
     Input: `str` or `np.ndarray` - Input fingerprint image.
     Output: `dict` - Probabilities for 'real' and 'synthetic'.
   - `detect_bonafide`
     Description: Detect if the given fingerprint is bonafide and not spoof.
     Input: `np.ndarray` - Input fingerprint image.
     Output: `bool` - True if bonafide, False otherwise.
7. **ACE-V Analysis:**
   api_name: `api_7`
   - `prepare_ace_sheet`
     Description: Prepare ACE-V style sheet with desired fields from a pair of fingerprint images.
     Input: Input fingerprint image.
     Output: `dict` - Output ACE-V sheet.
   - `compare_fingerprint_ace`
     Description: Get 'individualization' or 'exclusion' decision from the given ACE-V sheet.
     Input: `dict` - ACE sheet.
     Output: `str` - Decision for comparison ('individualization' or 'exclusion').
8. **Fingerprint Enhancement:**
   api_name: `api_8`
   - `enhance_image`
     Description: Enhance given fingerprint image for feature extraction.
     Input: `str` or `np.ndarray` - Input fingerprint image.
     Output: `np.ndarray` - Enhanced fingerprint image.
9. **Fingerprint Segmentation:**
   api_name: `api_9`
   - `segment_palm_print`
     Description: Segment the given palm print image into five fingerprint segments.
     Input: `str` or `np.ndarray` - Input palm print image.
     Output: Five `np.ndarray` - Five fingerprint images, one for each finger. Returns `None` if no fingerprint detected.
10. **Fingerprint Quality:**
    api_name: `api_10`
    - `get_quality_score`
      Description: Get quality score for the given fingerprint image.
      Input: `str` or `np.ndarray` - Input fingerprint image.
      Output: `int` - Integer quality score from 1 to 100.

Guidelines for Generating Questions:

- **Scenario Realism:** Design questions reflecting realistic application scenarios where multiple APIs must be used

in sequence or combined to achieve the correct outcome. Each question should require 3–5 function calls.
- **Functional Complexity:** Ensure each question involves varied functions across multiple APIs without relying on the same set of functions every time.
- **Logical Flow:** Each question should suggest a sequence that logically flows with the task requirements. Clarify steps needed for functions that build upon each other to reach the final answer.

Guidelines for Generating Options:
- **Complete API Chains:** Provide four option chains, each specifying a complete sequence of API function calls in the correct order. One sequence should be correct; the others should be logically incorrect but plausible.
- **Logical Plausibility of Distractors:** Distractors should appear reasonable and require reasoning to eliminate.
- **Randomized Answer Positioning:** Shuffle options so the correct answer appears randomly in position A, B, C, or D.

Example Question: In an airport security system, a fingerprint is enhanced and checked for bonafide print and, if yes, verified against the stored database. Which API sequence should be applied?

A. `api_8-enhance_image, api_2-extract_features, api_6-detect_bonafide, api_2-identify_fingerprint`
B. `api_8-enhance_image, api_2-extract_features, api_2-identify_fingerprint, api_6-detect_bonafide`
C. `api_8-enhance_image, api_6-detect_bonafide, api_2-extract_features, api_2-identify_fingerprint`
D. `api_6-detect_bonafide, api_2-extract_features, api_2-identify_fingerprint, api_8-enhance_image`

Correct Answer: `C. api_8-enhance_image, api_6-detect_bonafide, api_2-extract_features, api_2-identify_fingerprint`

References

[1] Anguli: Synthetic fingerprint generator. 1

[2] Shuai Bai et al. Qwen3-vl technical report. *arXiv*, 2025. 4

[3] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 1

[4] Gregory Fiumara, Patricia Flanagan, Matthew Schwarz, Elham Tabassi, and Christopher Boehnen. Nist special database 301. *Gaithersburg, MD, USA*, 2018. 1

[5] Gregory P Fiumara, Patricia A Flanagan, John D Grantham, Kenneth Ko, Karen Marshall, Matthew Schwarz, Elham Tabassi, Bryan Woodgate, and Christopher Boehnen. Nist special database 302: Nail to nail fingerprint challenge. 2019. 1

[6] Steven A Grosz and Anil K Jain. Universal fingerprint generation: Controllable diffusion model with multimodal conditions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(2):1028–1041, 2024. 1

[7] Dario Maio, Davide Maltoni, Raffaele Cappelli, James L. Wayman, and Anil K. Jain. Fvc2000: Fingerprint verification competition. *IEEE transactions on pattern analysis and machine intelligence*, 24(3):402–412, 2002. 1

[8] Dario Maio, Davide Maltoni, Raffaele Cappelli, James L Wayman, and Anil K Jain. Fvc2002: Second fingerprint verification competition. In *2002 International conference on pattern recognition*, pages 811–814. IEEE, 2002. 1

[9] Dario Maio, Davide Maltoni, Raffaele Cappelli, Jim L Wayman, and Anil K Jain. Fvc2004: Third fingerprint verification competition. In *International conference on biometric authentication*, pages 1–7. Springer, 2004. 1

[10] Kartik Narayan, VS Vibashan, and Vishal M Patel. Facexbench: Evaluating multimodal llms on face understanding. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2026. 2

[11] NIST. Swgfast: Document #9 standard for the documentation of analysis, comparison, evaluation, and verification (ace-v) in tenprint operations (tenprint). https://www.nist.gov/system/files/documents/2016/10/26/swgfast_standard-documentation-ace-v-tenprint_2.0_121124.pdf, 2016. Accessed: 2025-06-30. 2

[12] OpenAI. Gpt-5 system card. Technical report, OpenAI, 2025. 1