Supplementary Materials: mEOL: Training-Free Instruction-Guided Multimodal Embedder for Vector Graphics and Image Retrieval

Contents

A. Human Study

To assess human perception of retrieval quality, we conducted a user study with 15 participants. Each participant was shown the retrieved image results from three models: (1) LLaVE (baseline), (2) our model with raster image, and (3) our model with raster image + SVG code for a given text query. Participants rated each result on a 1 to 5 scale based on semantic alignment with the query. As shown in Fig 1, our models consistently outperform the baseline. In particular, incorporating structural SVG information yields further improvements over the raster-only setting.

Structure-Aware Semantics To assess how well each model retrieves icons that are not only semantically relevant but also structurally aligned with the input query, we conducted an additional human evaluation. Fig 2 show that participants rated the retrieved icons on a 1–5 scale based on how well each result captured both the meaning and structural resemblance of the textual query.

We observe that *LLaVE* consistently received lower ratings, indicating difficulty in capturing both semantic and structural cues. In contrast, *Ours (Raster Image)* achieved moderately higher ratings across most queries, while *Ours (Raster Image + SVG Code)* received the most favorable scores with a clear concentration in the 4–5 range. This result highlights the effectiveness of incorporating SVG structural signals, leading to more faithful icon retrieval in terms of both meaning and visual form.

Consistency of Human Preferences To further analyze how users evaluated retrieval outputs across models, we computed Spearman's rank correlation (ρ) between rating vectors for each participant. Fig 3 shows the density distributions of these correlations. We observe low correlation between *LLaVE* and either of our models, reflecting significant variation in how participants judged the baseline compared to ours. Notably, the correlation between our two models (*Raster Image* and *Raster Image + SVG code*) remains moderately high ($\rho \approx 0.55$ on average), indicating that participants not only

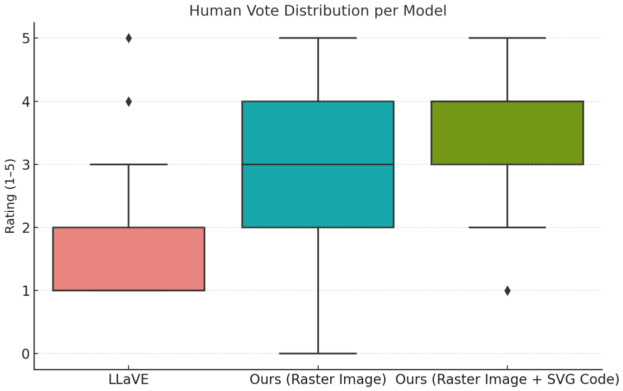


Figure 1. **User preference voting results.** For each query, participants were shown all three model outputs and asked to rate which result was most faithful to the intended semantics and visual impression.

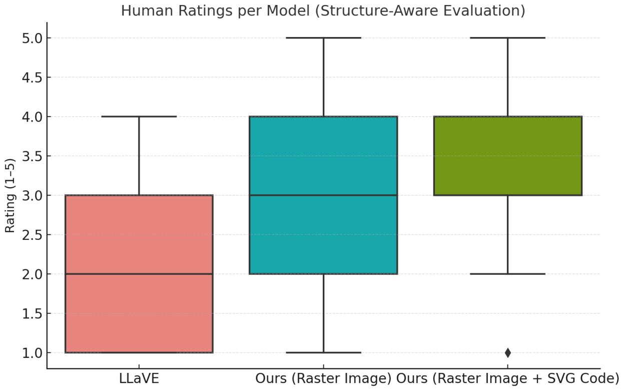


Figure 2. **Structure-aware evaluation ratings.** Participants rated how well each retrieved icon matched both the semantic meaning and structural characteristics of the input text.

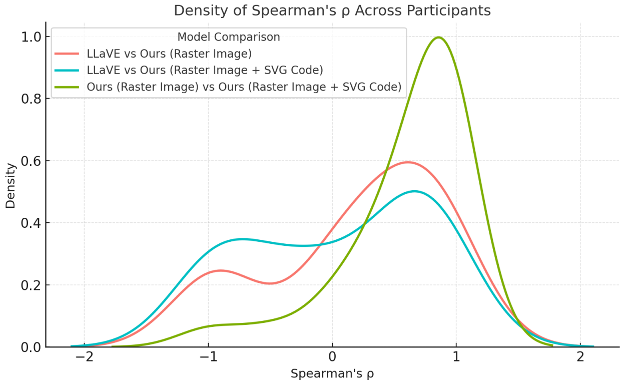


Figure 3. **Density plot of Spearman's rank correlation (ρ) computed across all participants for each pair of model outputs.** High correlation between *Ours (Raster Image)* and *Ours (Raster Image + SVG code)* suggests that the inclusion of SVG structure improves perceived quality while preserving relative rankings of retrieved results.

Table 1. **Ablation of Text Augmentation.** We analyze the change in retrieval performance when applying text augmentation, which adds information about incorrect answer choices to the original text query. We find that performance significantly decreases when text augmentation is applied.

Model	Text Augmentation	Recall@1	Recall@5	Recall@10
LLaMA-3.2-11B	✗	**0.3387**	**0.5785**	**0.6526**
	✓	0.3081	0.5276	0.6192
Qwen2.5-VL-7B	✗	**0.3503**	**0.6177**	**0.6846**
	✓	0.2413	0.4695	0.5567

consistently rated our models above the baseline but also preserved a stable preference for the SVG-enhanced variant across different queries.

B. Additional Experimental Result

Ablation of Text Augmentation. Each sample in VG-Bench [6] consists of a textual question, four textual answer choices, and a correct answer label. Currently, we use only the textual question and a correct answer label to construct the query. Through text augmentation, we also provide information from the incorrect answer choices. For example, if the original query is `"The answer is sadness"`, the text-augmented query would be `"The answer is sadness. Joy, surprise, anger are not the correct answers."` We hypothesize that by applying text augmentation, the increased amount of information can help the model to generate more semantic embeddings. However, as shown in Table 1, we find that applying text augmentation leads to a drop in performance. The additional details may cause a model that generates text autoregressively to move further away from the intended answer. This suggests that text augmentation is not a suitable approach for our method. These results imply that a simpler form of text, containing only the essential information needed for the query, may be more suitable for our method.

Table 2. **Embedding extraction time.** We measure the time required to construct the database by extracting embeddings from raster images and SVG codes using the Qwen2.5-VL-7B model, as well as the time computing query embeddings from text inputs.

Model	Input	Avg. Token Count	Avg. Time (s)
Qwen2.5-VL-7B	Image+SVG	1465.77	0.1161
	Text	81.82	0.0243

Embedding extraction time. We measure the time required to extract embeddings and report the results in Table 2. We observe that extracting embeddings from long original SVG codes and images takes approximately 0.11 seconds per sample, while text queries require about 0.02 seconds. Since the database embeddings are computed once and reused thereafter, similar to RAG pipelines, the overall runtime overhead remains minimal.

Qualitative Result. At Fig. 4, we provide more qualitative results. As shown, our embedding method can encode both structural part and the textual semantic meaning, compared to other methods such as CLIP [3], LLaVE [2].

C. Modality-Specific Prompting for mEOL

We compare each EOL method in Table 3, and additionally provide our modality-specific EOL prompts. For each modality, the prompt is given to effectively guide the model's understanding according to its modality, with SVG-specific instructions included to extract semantically meaningful embeddings, and the text and image include the visual semantics such as shapes, color, structure or layout.

D. Semantic SVG Code Enhancement Example

In Figure 5, we provide additional qualitative results of enhanced semantic SVG code. Original SVG codes typically consist of path components, which are sequences of numeric values representing lines, but they also contain numerous retrieval-irrelevant strings and formatting artifacts that introduce substantial noise. Providing such raw SVG code directly can make it challenging for a zero-shot embedder to generate appropriate embeddings. By leveraging vision-expert models (e.g., ChatGPT), we can enhance the structure of the code into a more interpretable form by assigning IDs and simplifying the overall structure. The enhanced SVG code becomes more interpretable, facilitating better multi-modal embeddings. For the enhanced SVG code generation, we adopted GPT-4o-mini with the instruction: *I provide you the SVG code: code and also the SVG image. Assign the ids(it must be one of the components of the object you guessed) to each components, and assign the object as the id of whole code. Then, please simplify the SVG code.*

Table 3. **Comparison with prior EOL-based methods.** Compared to the previous EOL-based methods, our method leverages modality-specific prompting and uses the penultimate layer.

Method	Prompt Template	Layer Index
PromptEOL [1]	This sentence: "[Prefix]" means in one word:	Last (-1)
KEEOL [5]	The essence of a sentence is often captured by its main subjects and actions, while descriptive terms provide additional but less central details.With this in mind, this sentence : "[Prefix]" means in one word:	Penultimate (-2)
GenEOL [4]	The essence of a sentence is often captured by its main subjects and actions, while descriptive terms provide additional but less central details.With this in mind, this sentence : "[Prefix]" means in one word:	Penultimate (-2)
Ours	**Text** (User): This textual description includes visual elements such as shapes, colors, structure, and layout. With this in mind, this sentence: "[X]" in one word. **Image** (User): "[X]". Closely observe the image|shapes, structure, colors, and details. With this in mind, this image means in one word. **SVG** (User): I provide you the SVG code: "[X]". What image does this SVG code make? Tell me in one word. **Image+SVG** (User): Below is an SVG icon and an image of the icon. "[X]""[X]". With this in mind, this icon means in one word.	**Penultimate** (-2)

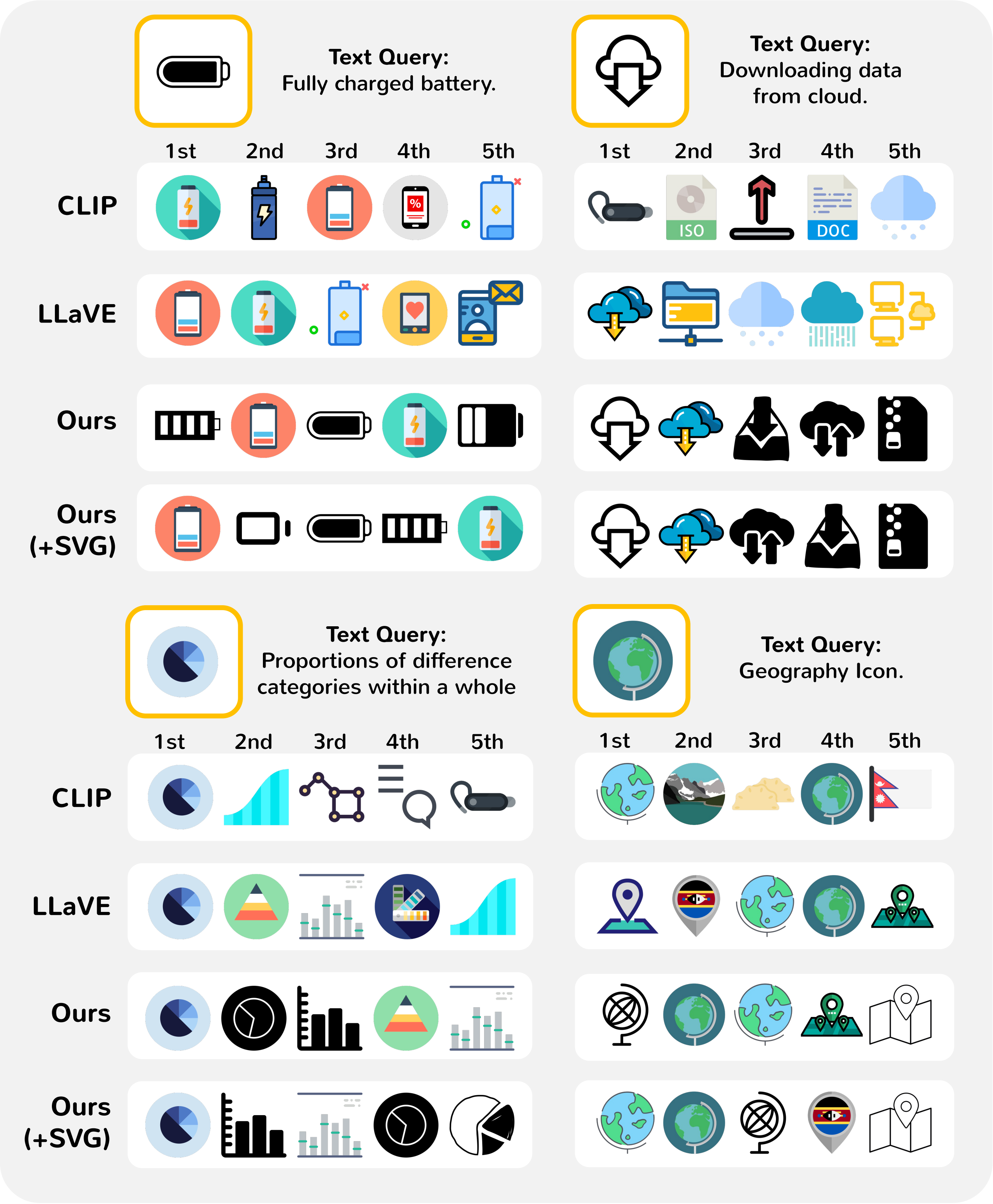
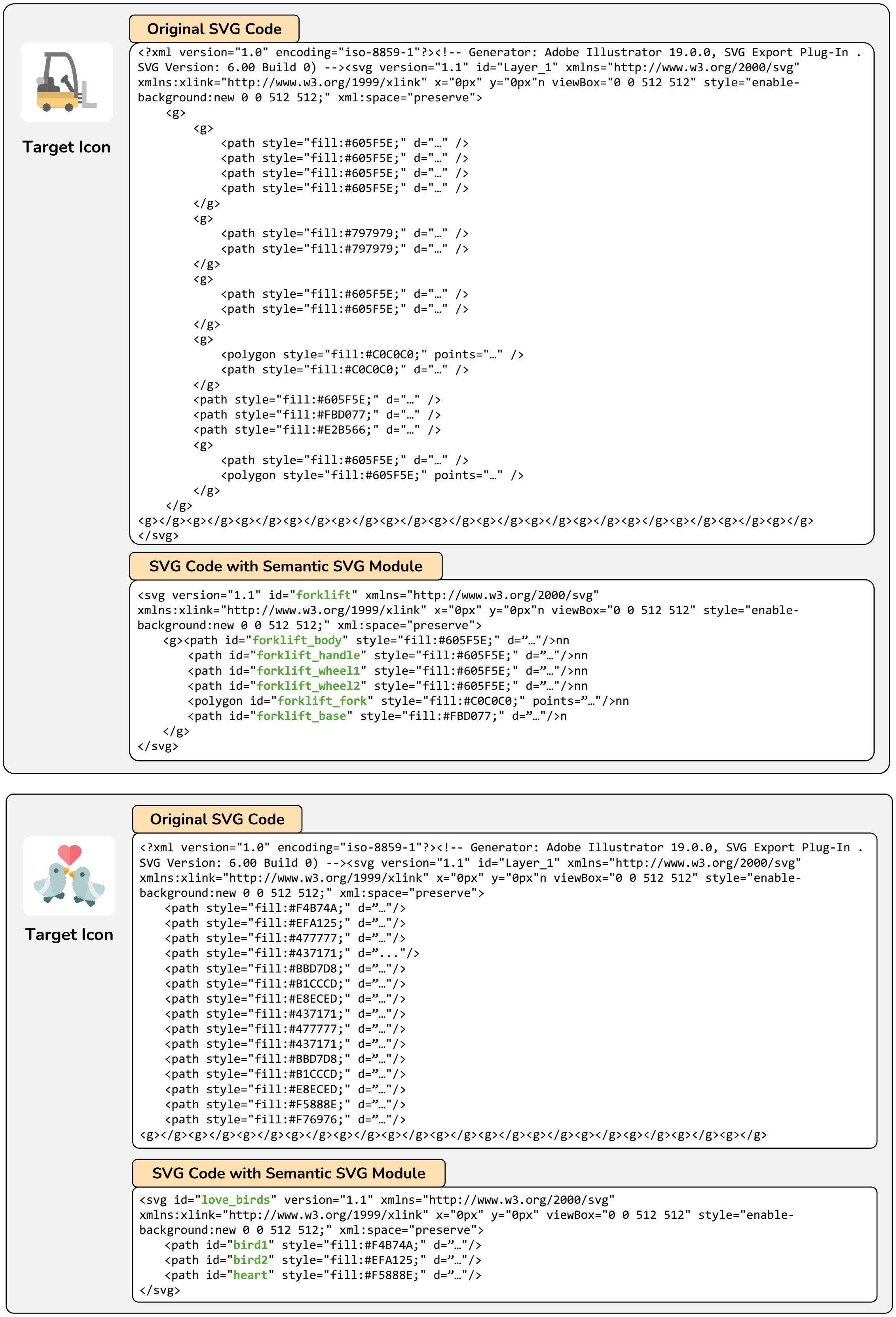


Figure 4. **Additional qualitative results.**



```
<?xml version="1.0" encoding="iso-8859-1"?><!-- Generator: Adobe Illustrator 19.0.0, SVG Export Plug-In .
SVG Version: 6.00 Build 0) --><svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px"n viewBox="0 0 512 512" style="enable-
background:new 0 0 512 512;" xml:space="preserve">
    <g>
        <g>
            <path style="fill:#605F5E;" d="…" />
            <path style="fill:#605F5E;" d="…" />
            <path style="fill:#605F5E;" d="…" />
            <path style="fill:#605F5E;" d="…" />
        </g>
        <g>
            <path style="fill:#797979;" d="…" />
            <path style="fill:#797979;" d="…" />
        </g>
        <g>
            <path style="fill:#605F5E;" d="…" />
            <path style="fill:#605F5E;" d="…" />
        </g>
        <g>
            <polygon style="fill:#C0C0C0;" points="…" />
            <path style="fill:#C0C0C0;" d="…" />
        </g>
        <path style="fill:#605F5E;" d="…" />
        <path style="fill:#FBD077;" d="…" />
        <path style="fill:#E2B566;" d="…" />
        <g>
            <path style="fill:#605F5E;" d="…" />
            <polygon style="fill:#605F5E;" points="…" />
        </g>
    </g>
<g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g>
</svg>
```

```
<svg version="1.1" id="forklift" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px"n viewBox="0 0 512 512" style="enable-
background:new 0 0 512 512;" xml:space="preserve">
    <g><path id="forklift_body" style="fill:#605F5E;" d=”…"/>nn
        <path id="forklift_handle" style="fill:#605F5E;" d=”…"/>nn
        <path id="forklift_wheel1" style="fill:#605F5E;" d=”…"/>nn
        <path id="forklift_wheel2" style="fill:#605F5E;" d=”…"/>nn
        <polygon id="forklift_fork" style="fill:#C0C0C0;" points=”…"/>nn
        <path id="forklift_base" style="fill:#FBD077;" d=”…"/>n
    </g>
</svg>
```

```
<?xml version="1.0" encoding="iso-8859-1"?><!-- Generator: Adobe Illustrator 19.0.0, SVG Export Plug-In .
SVG Version: 6.00 Build 0) --><svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px"n viewBox="0 0 512 512" style="enable-
background:new 0 0 512 512;" xml:space="preserve">
    <path style="fill:#F4B74A;" d=”…"/>
    <path style="fill:#EFA125;" d=”…"/>
    <path style="fill:#477777;" d=”…"/>
    <path style="fill:#437171;" d=”..."/>
    <path style="fill:#BBD7D8;" d=”…"/>
    <path style="fill:#B1CCCD;" d=”…"/>
    <path style="fill:#E8ECED;" d=”…"/>
    <path style="fill:#437171;" d=”…"/>
    <path style="fill:#477777;" d=”…"/>
    <path style="fill:#437171;" d=”…"/>
    <path style="fill:#BBD7D8;" d=”…"/>
    <path style="fill:#B1CCCD;" d=”…"/>
    <path style="fill:#E8ECED;" d=”…"/>
    <path style="fill:#F5888E;" d=”…"/>
    <path style="fill:#F76976;" d=”…"/>
<g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g><g></g>
```

```
<svg id="love_birds" version="1.1" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px" viewBox="0 0 512 512" style="enable-
background:new 0 0 512 512;" xml:space="preserve">
    <path id="bird1" style="fill:#F4B74A;" d=”…"/>
    <path id="bird2" style="fill:#EFA125;" d=”…"/>
    <path id="heart" style="fill:#F5888E;" d=”…"/>
</svg>
```

Figure 5. **Qualitative results of semantic SVG module.**

References

[1] Ting Jiang, Shaohan Huang, Zhongzhi Luan, Deqing Wang, and Fuzhen Zhuang. Scaling sentence embeddings with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3182–3196, 2024. 3

[2] Zhibin Lan, Liqiang Niu, Fandong Meng, Jie Zhou, and Jinsong Su. Llave: Large language and vision embedding models with hardness-weighted contrastive learning. *arXiv preprint arXiv:2503.04812*, 2025. 2

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

[4] Raghuveer Thirukovalluru and Bhuwan Dhingra. Geneol: Harnessing the generative power of llms for training-free sentence embeddings. *arXiv preprint arXiv:2410.14635*, 2024. 3

[5] Bowen Zhang, Kehua Chang, and Chunping Li. Simple techniques for enhancing sentence embeddings in generative language models. In *International Conference on Intelligent Computing*, pages 52–64. Springer, 2024. 3

[6] Bocheng Zou, Mu Cai, Jianrui Zhang, and Yong Jae Lee. Vgbench: Evaluating large language models on vector graphics understanding and generation. *CoRR*, 2024. 2