# Evolutionary Optimization of Model Merging Recipes

**Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, David Ha**
Sakana AI
Tokyo, Japan
{takiba,mkshing,yujintang,qisun,hadavid}@sakana.ai

## Abstract

We present a novel application of evolutionary algorithms to automate the creation of powerful foundation models. While model merging has emerged as a promising approach for LLM development due to its cost-effectiveness, it currently relies on human intuition and domain knowledge, limiting its potential. Here, we propose an evolutionary approach that overcomes this limitation by automatically discovering effective combinations of diverse open-source models, harnessing their collective intelligence without requiring extensive additional training data or compute. Our approach operates in both parameter space and data flow space, allowing for optimization beyond just the weights of the individual models. This approach even facilitates cross-domain merging, generating models like a Japanese LLM with Math reasoning capabilities. Surprisingly, our Japanese Math LLM achieved state-of-the-art performance on a variety of established Japanese LLM benchmarks, even surpassing models with significantly more parameters, despite not being explicitly trained for such tasks. Furthermore, a culturally-aware Japanese VLM generated through our approach demonstrates its effectiveness in describing Japanese culture-specific content, outperforming previous Japanese VLMs. This work not only contributes new state-of-the-art models back to the open-source community, but also introduces a new paradigm for automated model composition, paving the way for exploring alternative, efficient approaches to foundation model development.[1]

## 1   Introduction

Model merging [15, 28], a recent development in the large language model (LLM) community, presents a novel paradigm shift. By strategically combining multiple LLMs into a single architecture, this exciting development has captured the attention of researchers due to its key advantage: it requires no additional training, making it an incredibly cost-effective approach for developing new models. This accessibility has fueled a surge in interest and experimentation with model merging. The Open LLM Leaderboard [20] is now dominated by merged models, showcasing its potential for democratizing foundation model development.

However, model merging is considered by many to be a form of black art or alchemy, relying on the model maker's intuition and instincts about model selection and merging recipes to create and refine a new model that performs well for a particular task. Furthermore, the model maker is often required to have some domain knowledge for the various different benchmark tasks. Given the large diversity of open models and benchmarks in the community, human intuition can only go so far, and we believe a more systematic approach for discovering new model combinations will take things much further.

We believe evolutionary algorithms will be able to discover more effective model merging solutions, and thus provide a path for automating the creation of more capable models. As a step towards this direction, in this work, we show that evolution can be employed to discover novel and unintuitive ways to merge various models to produce new models with a new combined ability. In this work, we present a methodology that leverages evolutionary algorithms to facilitate the merging of foundation models. Our approach is distinguished by its ability to navigate both parameter space (weights) and the data flow space (inference path), proposing a framework that integrates these two dimensions.

---

[1] EvoLLM-JP, EvoVLM-JP release: `https://github.com/SakanaAI/evolutionary-model-merge`

This work makes several key contributions to the field of foundation model development:

1. **Automated Model Composition**: We introduce **Evolutionary Model Merge**, a general evolutionary method to automatically discover optimal combinations of diverse open-source models for creating new foundation models with user-specified capabilities. This approach harnesses the collective intelligence of existing open models, enabling the creation of powerful models without the need for extensive training data or compute.

2. **Cross-Domain Merging**: We demonstrate that our method can discover novel ways to merge models from disparate domains (e.g., non-English language and Math, non-English language and Vision), potentially exceeding the capabilities achievable through conventional human design strategies.

3. **State-of-the-Art Performance**: We showcase the effectiveness of our method by automatically generating a Japanese LLM with Math reasoning capability and a Japanese Vision-Language Model (VLM). Notably, both models achieve state-of-the-art performance on various benchmarks, even without explicit optimization for those tasks.

4. **High Efficiency and Surprising Generalizability**: We observe that our 7B parameter LLM surpasses the performance of some previous 70B parameter Japanese LLMs on benchmark datasets, highlighting the high efficiency and surprising generalization capability of our approach. We believe this model can serve as a strong general-purpose Japanese LLM.

5. **Culturally-Aware VLM**: The generated Japanese VLM achieves top results when tested on a domestically-sourced dataset of Japanese image-description pairs, demonstrating its ability to handle Japanese culture-specific content.

We are committed to open science and are excited to open-sourcing our EvoLLM-JP and EvoVLM-JP, two state-of-the-art Japanese foundation models, to the community, enabling further research and development in the field. Our work challenges the conventional paradigm of expensive model development by demonstrating that our evolutionary-based method can produce competitive models without relying on gradient-based training. This paves the way for exploring alternative, potentially more efficient, approaches to foundation model development.

## 2 Background and Related Work

### 2.1 Overview of Model Merging

Model merging offers a novel approach to leverage the strengths of multiple pre-trained models. It allows us to combine task-specific models, each potentially fine-tuned for a particular downstream task, into a single unified model. This technique stands in contrast to traditional transfer learning, where a pre-trained model is further fine-tuned for a new task. While transfer learning offers advantages like improved performance and faster convergence, the resulting models are typically limited to single tasks. Model merging, on the other hand, strives to create a versatile and comprehensive model by combining the knowledge from multiple pre-trained models, potentially yielding a model capable of handling various tasks simultaneously.

A simple method of merging multiple models is to average the weights of multiple models fine-tuned from the same base initial model. This *model soup* approach [48] demonstrated significant improvements on relatively large image processing and image classification models. The work pointed to similarity between weighted model averaging leading to a flat local minima. Theoretical [11, 25, 26, 37] and empirical studies dating back from the 1990s [18, 19] demonstrated that flatter local optima generalize better to out-of-distribution shifts. More recent work [8, 34] examined the theoretical properties and practical implications of weight averaging applied to larger generative models.

Linear weight averaging not only works well for image processing and classification models, but is also effective for image generation models such as latent diffusion models. The release of Stable Diffusion [39] led to a variety of specialist fine-tunes of the model for a variety of different styles, from photorealism, to anime, to realistic painting styles. Popular open-source WebUIs [4] enabled hobbyists to experiment with merging different models using linear or spherical linear interpolation (SLERP) [47] of weight parameters of distinct fine-tuned Stable Diffusion models. This has led to the discovery of merged models that combine strengths of specialized models into a single model. For some time, the most popular Stable Diffusion models were neither the original base models nor

the fine-tuned versions, but rather the merged models created by enthusiasts. This trend persists until the release of a more advanced base model, at which point the community's cycle of fine-tuning and merging is renewed.

## 2.2 Merging Language Models

Simple weight interpolation works well for merging image generation models that operate in a real valued latent and output space, but to our surprise, they also still work for merging language models under the right conditions, but have performance issues [49]. Recently, new methods have been proposed to address merging language models specifically. One method, Task Arithmetic [21], involves building *task vectors* by subtracting pre-trained from fine-tuned model weights, enabling manipulation through arithmetic operations to steer the merged model's behavior.

A key problem of weight interpolation-based methods is that they neglect parameter interference, leading to performance degradation. A recent work [49] identifies two key sources of interference: redundant parameter values and conflicting parameter signs across models, and proposes a method that addresses these issues to achieve improved merging performance. The proposed TIES-Merging method addresses information loss in existing merging methods by incorporating three steps: resetting minimal parameter changes, resolving sign conflicts, and merging only aligned parameters.

Another recent work [50] proposes the DARE method goes further by zeroing out small differences between the fine-tuned model and the original base model, while amplifying the differences. In practice, DARE [50] is often used together with Task Arithmetic [21] or TIES-Merging [49].

Image generation model merging really took off in the community only after such methods were implemented in open-source toolkits [4]. Similarly, language model merging really gained popularity after the implementation of *mergekit* [15, 28]. This toolkit provides all of the popular *recipes* for merging language models. In addition to simple linear and spherical interpolation, advanced recipes such as Task Arithmetic, TIES-Merging, DARE are also implemented, offering users the ability to experiment with combining these methods to work with fine-tunes of popular base models such as Mistral [22]. With this toolkit, a large number of capable merged models are developed by the community, and most of the top models on the Open LLM Leaderboard [20] are gradually dominated by merged models produced by the community of language model enthusiasts.

Mergekit also introduced an additional method, called *Frankenmerging* which is not based on weight merging, but for the user to experiment with trying to stack different layers from multiple models to sequentially create a new model. This method has the advantage of not tying the user to one particular family of models with a fixed architecture for merging (e.g. Mistral-based models), but potentially creating *new* architectures from merging entirely different models. It is noted that discovering new *Frankenmerging* technique remains a challenge for the community, and requires much more trial and error to discover new recipes for this technique. To date, almost everyone uses a similar *Frankenmerging* merge recipe, and there has been very little trial and error to improve upon it. The field remains highly under-explored, and we believe this is where evolution can help.

## 2.3 Connection to Evolutionary Neural Architecture Search

Model merging holds significant potential and democratizes the model-building process, making it accessible to a broad range of participants. However, it heavily relies on intuition and domain knowledge. Human intuition, however, has its limits. With the growing diversity of open models and tasks, we need a more systematic approach. We believe evolutionary algorithms, inspired by natural selection, can unlock more effective merging solutions. These algorithms can explore a vast space of possibilities, discovering novel and counter-intuitive combinations that traditional methods and human intuition might miss.

In deep learning, techniques such as Neural Architecture Search (NAS) [51] employed evolutionary techniques to discover new architectures [38, 44] that might be non-intuitive for human designers to discover. However, NAS-methods consumed significant computational resources given how each candidate model architecture needed to be trained. Our approach aims to *save* compute resources by discovering structures that take advantage of existing capabilities of existing Transformer blocks that had already consumed considerable resources to train in the first place.

Pre-trained transformer blocks are akin to neural network components one can choose to mix and match using an evolutionary architecture search algorithm. However, unlike NAS, we do not need to

train the model, and can evaluate the candidates right away. In a sense, our work resonates with the goals of earlier works of morphology search such as NEAT [45] and more recent work such as Weight Agnostic Neural Networks [12] which evolved neural network structures with certain task-specific inductive biases, without the need to train the weight parameters using gradient descent. Similarly, this was how NAS methods such as SMASH [45] avoided costly inner-loop training through the application of a Hypernetwork [16] to estimate the weights of architectural candidates.

In this work, we apply evolution to not only automate the optimization of model merging recipes in weight space, confined to a single architecture, but also explore applying evolution to optimize the stacking of layers from different models—a more involved approach that has the potential to create entirely novel neural architectures from existing building blocks. Unlike weight merging optimization which confine space of models to be merged together to fine-tuned descendents of the same parent base model, layer stacking optimization has no such constraints. Our work systematically explores the application of evolution applied in both parameter and layer space.

# 3 Method

Our goal is to create a unified framework capable of automatically generating a merged model from a selection of foundation models, ensuring that the performance of this merged model surpasses that of any individual in the collection. Central to our approach is the application of evolutionary algorithms, which we employ to refine the intricacies involved in model merging. To systematically address this challenge, we first dissect the merging process into two distinct, orthogonal configuration spaces, analyzing their individual impacts. Building on this analysis, we then introduce a cohesive framework that seamlessly integrates these spaces. Figure 1 provides a schematic representation of our approach.
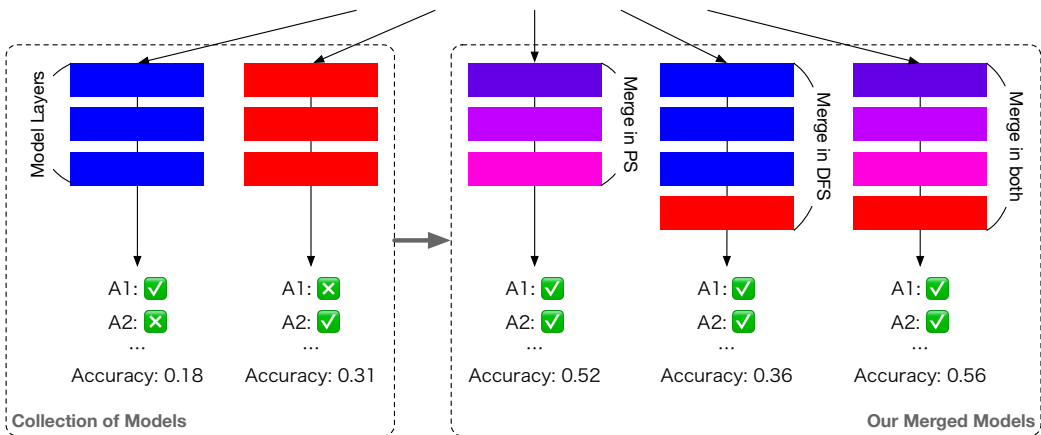


Figure 1: **Overview of Evolutionary Model Merge**. Our approach encompasses (1) evolving the weights for mixing parameters at each layer in the parameter space (PS); (2) evolving layer permutations in the data flow space (DFS); and (3) an integrated strategy that combines both methods for merging in both PS and DFS. Notice that merging in the PS is not simple copying and stitching of the layers parameters, but also mixes the weights. This merging is akin to blending colors as illustrated here (e.g., red and blue becomes purple). Note that we translated the questions to English for the reader; the models operate on Japanese text.

## 3.1 Merging in the Parameter Space

Model merging in the parameter space (PS) aims to integrate the weights of multiple foundational models into a unified entity with the same neural network architecture, yet outperforming the individual models. While various strategies for combining model parameters exist [34, 48], our approach leverages task vectors analysis to understand each model's strengths, based on the specific tasks they are optimized for or excel in [21]. Specifically, we enhance TIES-Merging with DARE [49, 50], allowing for more granular, layer-wise merging (In this paper, by "layer" we mean the input/output

embedding layers or a transformer block). We establish merging configuration parameters for sparsification and weight mixing at each layer, including input and output embeddings. These configurations are then optimized using an evolutionary algorithm, such as CMA-ES [17], for selected tasks, guided by critical task-specific metrics (e.g., accuracy for MGSM, ROUGE score for VQA).

## 3.2 Merging in the Data Flow Space

Recent analysis and discoveries imply that knowledge is stored distributedly in language models [14, 35, 36], suggesting simple yet novel model merging possibilities in the data flow space (DFS). Unlike merging in PS, model merging in DFS preserves the original weights of each layer intact. Instead, it optimizes the inference path that tokens follow as they traverse through the neural network. For example, after the $i$-th layer in model $A$, a token may be directed to the $j$-th layer in model $B$.

In our initial effort in this domain, we limit ourselves to serial connections and non-adaptive configurations, deferring the investigation of more flexible model merging to future work. Concretely, with a collection of $N$ models and a budget $T$, our method searches for a sequence of layer indices $L_{i,j}^{(t)}$ that delineates the path all the tokens should follow for a specific task. Here $L_{i,j}$ denotes the $j$-th layer in the $i$-th model, with $t \in [1, T]$ marking the step in the inference path.

One can quickly imagine how large the search space is. Assuming the total number of layers across all models is $M$, the size of the search space is $(M + 1)^T$, here the extra one indicates the inclusion of a pass-through layer. Even with a modest setting of $M = 64$ (e.g., 2 models of 32 layers each) and $T = 60$, this setting translates to an astronomically large search space, a challenge even for a capable evolutionary search algorithm. Luckily, our preliminary studies indicated that certain layer arrangements, particularly repetitive or permuted sequences from earlier in the model, can adversely affect performance. Based on this, we modify our settings to include an indicator array $\mathcal{I}$ of size $T = M \times r$ in the evolutionary search space, here $r$ is number of repetitions.

Conceptually, we layout all the layers in sequential order (i.e., all layers in the $i$-th model followed by those in the $i + 1$-th model) and repeat them $r$ times, the indicator array then manages the inclusion/exclusion of layers. If $\mathcal{I}_i > 0$ we include the layer corresponding to index $i$ in the slots in the merged model, otherwise we exclude it. Consequently, our search space is reduced to $2^T$, which is still large, but tractable for evolutionary search.

In our search, we only optimize the data inference path inside the merged model and keep parameters in the models intact. In this setting, a layer may face an input whose distribution is different from what it is used to (from its original model), leading to unexpected outputs. For example, our preliminary studies show that swapping a pair of neighboring layers in a language model makes its performance drop. Although more theoretical studies are needed to model the distribution shift, empirically we find that appropriately scaling an input that wishes to go from layer $i$ to $j$ by $W_{ij}$ help alleviate the problem. Here, $W \in \mathcal{R}^{M \times M}$ is a matrix that is also optimized by the evolutionary search together with the indicator array $\mathcal{I}$.

The size of $W$ grows quadratically with $M$, for scenarios involving a large number of layers. An alternative approach to contain the search space size involves parameterizing $W$ with a neural network [16, 40]. We can instead evolve a feed-forward network to output the scaling weights conditioned on the layer and step indices: $W_{ij} = \pi_\theta(i, j, t)$ where $\theta$'s are the parameters to be evolved, whose size does not change when $M$ grows.

## 3.3 Merging in Both Spaces

Model merging in the PS and in the DFS are orthogonal approaches, however, it is straightforward to combine these disentangled methods and further boost the performance of a merged model. As we show in the rightmost illustration in Figure 1 and in Section 4.1, it is possible to first apply PS merging to a collection of models, and then put back this merged model in the collection and apply DFS merging from this enlarged collection.

This can be extremely helpful when one considers model merging with multiple objectives, wherein PS merging can be applied first to produce several merged models each of which targets one of the muliple objectives of interest, and then DFS merging is applied with multi-objective genetic algorithms such as NSGA-II [10] to further expand the final model's performance in relevant metrics.

# 4 Experiments

Most merged models in the community optimize for a narrow set of tasks defined in The Open LLM Leaderboard [20]. Our motivation is for evolutionary search to discover novel ways to merge different models from vastly different domains (e.g., non-English language and Math, or non-English language and Vision) which might be difficult for human experts to discover effective merging solutions themselves. Furthermore, effectively merging models from very different domains can lead to models of wider real-world applicability and enable us to develop models beyond the large population of models that are optimized for the narrow range of tasks defined by a leaderboard.

We demonstrate our Evolutionary Model Merge approach described in Section 3 by evolving a Japanese LLM capable of Math reasoning, and a Japanese VLM proficient in handling culturally-specific content. Specifically, in Section 4.1, we apply evolution to merge a Japanese LLM with an English Math LLM to build a Japanese Math LLM, and in Section 4.2, we apply evolution to merge a Japanese LLM with an English VLM to create a Japanese VLM.

## 4.1 Evolving Japanese Math LLM

### 4.1.1 Setup

**Source Models**  To develop a model capable of solving math problems in Japanese, we apply evolutionary model merge on a set of source models containing a Japanese LLM and Math LLMs: `shisa-gamma-7b-v1` [3] (Japanese LLM), `WizardMath-7B-V1.1` [33] and `Abel-7B-002` [6]. All these models are fine-tuned from `Mistral-7B-v0.1` [22].

**Dataset**  For testing, we used the MGSM dataset [41], a multilingual translation of a subset of the GSM8k dataset [7]. The Japanese test set of MGSM, consisting of 250 samples, was used for the final evaluation. We used a different dataset for evolutionary search to avoid overfitting the test set. Specifically, we translated the remaining 1069 samples (out of 1319 examples) of the GSM8k test set that were not included in the MGSM test set into Japanese.[2]

**Evaluation**  We evaluated the ability to generate Japanese answers to Japanese math problems. Therefore, we considered an answer correct if it met the following criteria: (1) the concluding numerical value must be correct, and (2) the reasoning text should be written in Japanese.

We treated the last numerical value appearing in the output as the answer. We needed to adopt this heuristic because we are merging multiple models that were trained in different formats, which made it difficult to correct the output format. This method appeared to extract the answers correctly in almost all cases. Additionally, to determine the language of the output, we utilized fasttext [23, 24]. We used greedy sampling for generation and calculated the zero-shot pass@1 accuracy.

**Optimization**  For optimization in PS, we used the CMA-ES[17] algorithm implemented in Optuna [2] with default hyper-parameters. Specifically, we set all initial parameter values to 0.5, sigma to 1/6, and the population size to $4 + \lfloor 3 \ln(n_{\text{params}}) \rfloor$, where $n_{\text{params}}$ is the number of parameters to optimize. The fitness value is defined as the accuracy for all 1069 training samples. Please note that this set is disjoint from MGSM's test set. The optimization was conducted for 1000 trials, and the best trial with respect to the training accuracy was chosen as the final model. We decided to employ TIES-Merging [49] with DARE [50] through preliminary experiments, and optimized its parameters.

In our DFS merging experiments, $M = 64, r = 3$, and consequently, $T = M \times r = 192$. We kept the last 200 examples in the training data as our validation set and optimize on the rest of the data with a batch size of 200. We report the performance of the snapshot that achieved the highest accuracy in the validation set, and the test set is strictly isolated from the optimization process. We adopted CMA-ES in EvoJAX [46], it optimized $\mathcal{I}$ and $W$ for a total of 100 generations with a popluation size of 128, and we used the default hyper-parameters. We limited our DFS merging to two models $A$ and $B$ to ensure that the final model remains modest in size and can be run on a single GPU, but in principle, the methodology can scale to merging multiple models. During the merging, model $A$'s

---

[2]One may think that it is more natural to translate the GSM8k training set. However, in our preliminary efforts, this approach did not work well. Since open-source math models were trained on the GSM8k training set, we were unable to perform accurate evaluations.

Table 1: **Performance Comparison of the LLMs.** Models 1–3 are source models, Models 4–6 are our optimized merge models, and Models 7–11 are provided for reference. MGSM-JA measures math ability in Japanese, and JP-LMEH evaluates general Japanese language capabilities, with a detailed breakdown provided in Table 2.

| Id. | Model | Type | Size | MGSM-JA (acc ↑) | JP-LMEH (avg ↑) |
|---|---|---|---|---|---|
| 1 | Shisa Gamma 7B v1 | JA general | 7B | 9.6 | 66.1 |
| 2 | WizardMath 7B v1.1 | EN math | 7B | 18.4 | 60.1 |
| 3 | Abel 7B 002 | EN math | 7B | 30.0 | 56.5 |
| 4 | **Ours (PS)** | 1 + 2 + 3 | 7B | **52.0** | **70.5** |
| 5 | **Ours (DFS)** | 3 + 1 | 10B | **36.4** | **53.2** |
| 6 | **Ours (PS+DFS)** | 4 + 1 | 10B | **55.2** | **66.2** |
| 7 | Llama 2 70B | EN general | 70B | 18.0 | 64.5 |
| 8 | Japanese StableLM 70B | JA general | 70B | 17.2 | 68.3 |
| 9 | Swallow 70B | JA general | 70B | 13.6 | 71.5 |
| 10 | GPT-3.5 | commercial | - | 50.4 | - |
| 11 | GPT-4 | commercial | - | 78.8 | - |

tokenizer, and input/output embeddings are utilized. Furthermore, to maintain compatibility with the embedding layers, we mandate that the initial and final transformer layers of model $A$ define the start and the end of the inference path. We initialized the indicator array $\mathcal{I}$ so that all layers in model $A$ are more likely to be included as initial hops in the inference path to shorten the search time.

### 4.1.2 Results

Table 1 summarizes the performance of the LLMs on Japanese Math and overall Japanese benchmark tasks. The MGSM-JA column reports the results from the MGSM test set, employing the previously described metrics. The Japanese language model (Model 1) demonstrates limited mathematical proficiency, while the Math models (Models 2 and 3), though mathematically adept, show insufficient command of the Japanese language. Consequently, all three models score low scores on the MGSM-JA, with accuracy at or below 30.0.

In contrast, our merged models (Models 4–6) manifest a substantial elevation in performance. Notably, the model merged in PS (Model 4) achieves an impressive score of 52.0, highlighting the remarkable potential in combining models with distinct expertises. The DFS-merged model (Model 5) also shows a performance enhancement, with an over 6 percent increase in accuracy compared to the source models. While the leap in performance is not as pronounced as with PS merging, it still proves to be a valid and orthogonal approach. Finally, our hybrid model (Model 6), which integrates both merging strategies, shows further enhancements on the task.

Figure 2 gives an overview of the five models' "answer sheet" on the math problems. Our merged models retain the foundational knowledge in the source models, as evidenced by the similar score patterns on problems 1–15. Moreover, they exhibit emergent capabilities, successfully tackling problems that stumped the source models (e.g., problems 20–30). Evidently, by effectively integrating a Japanese LLM and mathematical models, we have succeeded in producing models that are proficient in both Japanese language understanding and mathematical problem-solving.
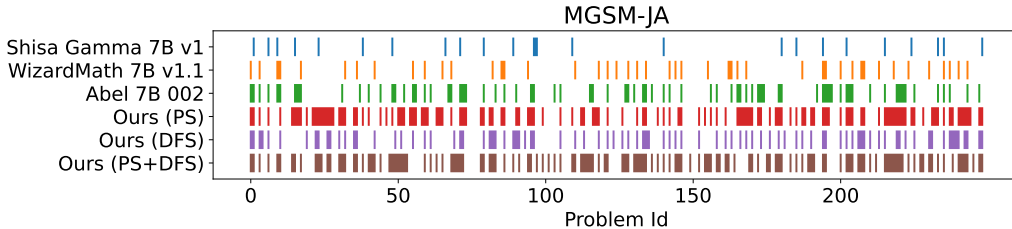


Figure 2: **Performance Overview.** The figure depicts the success of various models on the MGSM-JA task, with each of the 250 test problems represented along the x-axis by problem ID. Correct answers are indicated by colored markers at the corresponding positions.

7

Furthermore, Table 2 presents the results of evaluating the general Japanese language ability using the Japanese Language Model Evaluation Harness (JP-LMEH) benchmark suite. This benchmark suite consists of nine tasks, and the average score across these tasks is widely used as an indicator of overall Japanese language proficiency. Our models achieve remarkably high scores of 70.5 and 66.2, surpassing the source models and even the previous state-of-the-art 70B parameter Japanese LLM (specifically, Japanese StableLM 70B), despite having only 7B to 10B parameters.

Compared to the source Japanese model (Shisa Gmma 7B v1), it is interesting to note that there is not only a significant improvement in the MGSM scores but also improvements in other tasks such as JSQuAD and JAQKET. It should be noted that the MGSM scores here do not match those in Table 1, due to the differences in evaluation protocols (few-shot, prompting, etc.). Additional details are provided in Appendix A, including extensive comparisons with other models in Table 4.

Moreover, Appendix C showcases intriguing examples that demonstrate the utility of our models merged using evolution. The merged models correctly answered questions that require both knowledge about Japanese-culture *and* Math ability. In contrast, even if such Japanese questions were translated into English and answered in English, English math models would likely fail to provide the correct answers as they may not be aware of Japanese culture-specific context in the questions.

Table 2: **Breakdown of JP-LMEH Scores for Japanese Language Proficiency**. JP-LMEH (Japanese Language Model Evaluation Harness) is a benchmark suite consisting of 9 tasks, and the average score (Avg column) is used as an indicator of overall Japanese language proficiency. Full results are in Table 4

| | | JP Language Model Evaluation Harness | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Model | Size | JComQA | JNLI | MARC | JSQuAD | JAQKET | XLSum | XWino | MGSM | JCoLA | Avg |
| Shisa Gamma 7b v1 | 7B | 91.2 | 72.1 | 94.6 | 73.9 | 68.0 | 25.9 | 80.5 | 29.6 | 58.7 | 66.1 |
| WizardMath 7B V1.1 | 7B | 74.7 | 42.7 | 90.4 | 84.6 | 68.5 | 22.3 | 69.8 | 38.8 | 48.9 | 60.1 |
| Abel 7B 002 | 7B | 70.3 | 51.8 | 62.3 | 83.8 | 69.0 | 22.5 | 68.2 | 28.0 | 52.7 | 56.5 |
| **Ours (PS)** | 7B | 89.1 | 65.7 | 95.4 | 89.5 | 77.7 | 25.5 | 81.2 | 50.0 | 60.5 | **70.5** |
| **Ours (DFS)** | 10B | 67.7 | 58.2 | 53.5 | 66.8 | 54.3 | 17.3 | 65.6 | 30.0 | 65.6 | **53.2** |
| **Ours (PS+DFS)** | 10B | 88.2 | 50.3 | 91.5 | 78.6 | 77.8 | 23.2 | 73.0 | 40.0 | 73.0 | **66.2** |
| Llama 2 70B | 70B | 80.2 | 53.4 | 94.4 | 91.6 | 80.1 | 21.8 | 73.6 | 30.4 | 54.6 | 64.5 |
| Japanese Stable LM 70B | 70B | 91.2 | 50.4 | 92.9 | 87.1 | 88.4 | 24.3 | 82.0 | 37.2 | 61.7 | 68.3 |
| Swallow 70B | 70B | 95.3 | 57.2 | 91.7 | 94.1 | 93.9 | 23.1 | 83.3 | 45.2 | 59.5 | 71.5 |

### 4.1.3 Analysis

In our exploration of model merging in the parameter space, we experimented with diverse configurations, such as varying the assignment of merging parameters across different layer groups. However, due to a constrained dataset, we did not witness notable improvements in performance correlating with increase in configuration complexity. Consequently, we focused our reporting on a PS merged model (Model 4 in Table 1) that adopts the simplest setting: considering each source model as a singular layer and allocating two DARE-TIES associated parameters to each for evolutionary merging. Figure 3 illustrates the evolved parameter configuration post PS merging.

The CMA-ES optimization results reveals that all three models are important, as suggested by the uniformity of the optimized weighting values. The dominant density from the Japanese LLM suggests its critical contribution to solving the task. We conjecture that this may also be partially attributed to the Japanese LM's larger amount of fine-tunings from the Mistral base model. In line with the discussion in [50, Section 4.6], the sparsification of DARE tends to degrade performance when applied to such extensively fine-tuned models. Our evolutionary search has seemingly managed to address this issue by increasing the density for the Japanese LM.

The parameter settings derived from our PS merging experiments align well with outcomes from our DFS merging efforts. By incorporating the PS-merged model into our pool of source models and applying DFS merging across all potential pairings, we observed optimal performance with the combination of the PS-merged model and the Japanese language mode (Model 6 in Table 1). This finding echoes the significant influence of the Japanese language model, as indicated by its notable presence in Figure 3, and reaffirms the substantial promise of evolutionary model merging.

Figure 4 displays the evolution of the inference path, where our approach consistently recognized the value of initial steps, incorporating every layer from the first model (our PS-merged model), except for the last decoding layer and the embedding layer. As the process advanced, the method refined the

selection to a smaller, more effective set of layers and strategically alternated between layers from both contributing models. Significantly, the scaling parameters $W_{ij}$ emerged as crucial elements, our ablation studies revealed that eliminating them in the evolved model (e.g., by setting $W_{ij} = 1$) led to a performance decline exceeding 20 percent, highlighting their importance in the model's efficacy.
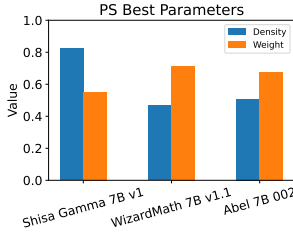


Figure 3: **Evolved Configurations for PS merging.** Although the weights are similar across the three source models, the pronounced density from the Japanese LLM underscores its pivotal role in our merged model.
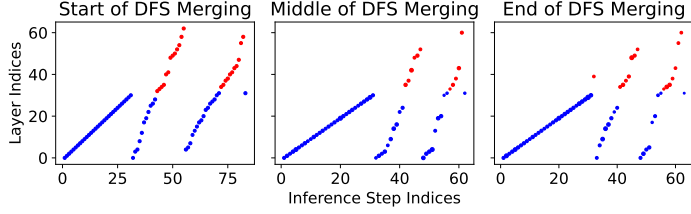
Figure 4: **Evolved Configurations for DFS Merging of models** $A$ **and** $B$**.** The three figures depict the evolution of the inference path on the MGSM-JA task. The y-axis represents the layer index $l \in [1, M]$, and the x-axis corresponds to the path index $t \in [1, T]$. Blue markers indicate path steps utilizing layers from model $A$, and red markers denotes those from $B$. Marker size reflects the magnitude of the scaling factor $W_{ij}$. The evolutionary search result includes most layers in $A$ at an early stage and then alternates between layers from both models. This result is from our 10B model (PS+DFS).

## 4.2 Evolving Japanese VLM

### 4.2.1 Multi-modality Extension

We now extend our method to multi-modal models, and evolve a culturally-specific content aware Japanese VLM. VLMs have recently shown remarkable progress by applying the powerful instruction-following capabilities of pre-trained LLMs. The architecture of a VLM generally consists of three components: (1) A vision encoder to extract image features; (2) An LLM to generate text (for the purpose of describing an image); and (3) A projection network to map image features into the LLM's embedding space [5, 9, 29, 30, 32]. Crucially, the LLM component is initialized with powerful pre-trained LLMs for their text generation capabilities. During training, the projection network and optionally the LLM are trained on various vision-language datasets, while the vision encoder is fixed.

### 4.2.2 Setup

**Source Models** The LLM component inside a VLM can be regarded as a standalone LLM, with the extra capability of understanding visual *soft prompts*. From this perspective, by fixing the vision encoder and the projection network and only focusing on the LLM component, it is straightforward to apply the methodologies detailed in Section 3 to produce a new LLM with expanded capabilities.

In this experiment, we merge a Japanese LLM and the LLM component in a VLM in the parameter space. We select `shisa-gamma-7b-v1` [3] as the Japanese LLM and `LLaVA-1.6-Mistral-7B` [31] as the VLM. Both models are fine-tunes of the `Mistral-7B-v0.1` [22] base model.

**Dataset** To the best of our knowledge, publically accessible Japanese VLM datasets are scarce. In response, we created a new open Japanese VLM benchmark and assessed our VLM on a widely recognized Japanese VQA dataset. Our new benchmark dataset consists of:

- **JA-VG-VQA-500:** A 500-sample test set extracted from the Japanese Visual Genome VQA dataset [42].
- **JA-VLM-Bench-In-the-Wild:** A Japanese version of LLaVA-Bench-In-the-Wild [32] . We compiled a rich collection of 42 images, accompanied by a total of 50 questions, featuring a variety of Japanese cultural elements and objects found in Japan. The QAs were crafted with the assistance of GPT-4V [1] and underwent a human-in-the-loop filtering process to eliminate nonsensical outcomes. Compared to the JA-VG-VQA-500 dataset, our set poses more complex challenges, demanding more nuanced and detailed responses.

We used another subset of the Japanese Visual Genome VQA dataset during the evolutionary search. This subset is not overlapped with examples in the JA-VG-VQA-500 dataset, to avoid leakage in the optimization process.

**Evaluation**  We consider two baselines in our experiments: `LLaVA-1.6-Mistral-7B` [31], one of our source models, and `Japanese Stable VLM` [43] a Japanese VLM trained from scratch on Japanese datasets.

All models adopt the same generation configurations, with deterministic decoding. We compute ROUGE-L with a Japanese language detector to replace non-Japanese responses with empty texts, resulting in a score of zero for non-Japanese responses. To be consistent with our LLM experiments in Section 4.1, we also employed fasttext [23, 24] for this language detection task. However, we made an exception for cases where the ground-truth answer itself contains non-Japanese but commonly seen words in Japanese texts (e.g., a widely recognized acronym such as "UFO"). In these instances, non-Japanese responses from models are not converted to empty texts.

**Optimization**  We use the identical settings as the earlier LLM PS-merging experiment in Section 4.1, and TIES-Merging with DARE for merging the source models in the parameter space.

### 4.2.3  Results

Table 3 compares the performance of our VLM with the baselines. Please note that the Japanese Stable VLM cannot be evaluated on JA-VG-VQA-500 because it was trained on this dataset.

Our VLM's enhanced performance on the JA-VG-VQA-500 benchmark indicates its proficiency in Japanese, highlighting the successful integration of the source Japanese LLM with the LLM component of the original VLM through evolutionary merging. Furthermore, our VLM's superior results on the JA-VLM-Bench-In-the-Wild compared to both baselines exhibits its adeptness at navigating culturally-specific content.

Besides the quantitative results in Table 3, we qualitatively compare our VLM with the baseline models in Appendix C. Our evolved model is able to handle Japanese culture-specific content remarkably well, generally producing more detailed responses with correct information.

Table 3: **Performance Comparison of the VLMs.** LLaVA 1.6 Mistral 7B is the source VLM and Japanese Stable VLM is an open-sourced Japanese VLM. While JA-VG-VQA-500 measures general VQA abilities in Japanese, JA-VLM-Bench-In-the-Wild evaluates the model's handling of complex VQA tasks within Japanese cultural contexts.

| Model | JA-VG-VQA-500 (ROUGE-L ↑) | JA-VLM-Bench-In-the-Wild (ROUGE-L ↑) |
|---|---|---|
| LLaVA 1.6 Mistral 7B | 14.3 | 41.1 |
| Japanese Stable VLM | - | 40.5 |
| **Ours** | **19.7** | **51.2** |

## 5  Discussion and Future Work

In this report, we propose a general method that uses evolutionary techniques to efficiently discover the best ways to combine different models from the vast ocean of different open-source models with diverse capabilities. By working with the vast collective intelligence of existing open models, our method is able to automatically create new foundation models with desired capabilities specified by the user. We find that our approach is able to automatically discover novel ways to merge different models from vastly different domains (e.g., non-English language and Math, or non-English language and Vision), in non-trivial ways that might be difficult for human experts to discover themselves.

To test our approach, we apply our method to automatically create a Japanese LLM capable of Math reasoning, and a culturally-specific content aware Japanese VLM. Surprisingly, we find that both models achieve state-of-the-art results on several LLM and Vision benchmarks, while not being explicitly optimized to be good at these benchmarks, attaining the top performance on a vast array

of other Japanese LLM benchmarks, even exceeding the performance of some previous SOTA 70B parameter Japanese LLMs.

With these promising initial results, we believe we are just scratching the surface of unlocking the full capabilities of evolutionary model merging, and this is the inception of a long-term development of applying evolutionary principles to foundation model development.

Currently, we are already achieving promising results in applying evolutionary model merging to image diffusion models, enabling the creation of high performance cross-domain image generation models by merging existing building blocks in novel ways discovered by evolution.

The method currently requires the user to select a set of source models to use as ingredients for evolutionary search. We believe it is also possible to leverage evolution to search for candidate source models from a vast population of existing models as well. In addition to model selection, we are also exploring using evolution to produce swarms of diverse foundation models each with its own niche and behaviors. This holds the potential of enabling the emergence of a collective intelligence consisting of a swarm of models capable of self-improvement by continuously producing new complementary internal models of the world through interaction.

Related to our work is an experiment, called Automerge [27], released at around the same time as this work. This interesting experiment works by selecting two random models from the top 20 models on the Open LLM Leaderboard [20] and randomly apply SLERP [47] or DARE-TIES [49, 50] to create new models. Over time, some of these models will do well, or even better on the benchmark tasks that define this leaderboard, becoming part of the leaderboard. We predict this approach will lead to combinations of the merged models that overfit to the benchmark tasks defined on the leaderboard. The author acknowledged that the idea behind this project was less about creating better models, but more about getting more metrics to help derive a more principled approach to model merging.

Our work takes an orthogonal approach of optimizing for tasks outside of the domain specified by the original leaderboard [20], rather than being confined by it. As we have shown, surprisingly, stepping away from optimizing for a particular benchmark occasionally results in even greater generalization to numerous other benchmark tasks that we had not intended to optimize for, and such emergent generalization might be the key to unlocking the next great advancements in AI.

The ability to evolve new models with new emergent capabilities, from a large variety of existing, diverse models with various capabilities have important implications. With the rising costs and resource requirement for training foundation models, by leveraging the rich variety of foundation models in the rich open-source ecosystem, large institutions or governments may consider the cheaper evolutionary approach for developing proof-of-concept prototype models quickly, before committing substantial capital or tapping into the nation's resources to develop entirely custom models from scratch, if that is even needed at all.

# 6   Limitations

We acknowledge that although our evolutionary model merging effectively integrates diverse expertise from the source models, it also inherits their limitations. For instance, we encountered instances where the merged models produced responses that lacked logical coherence. Additionally, this study does not encompass instruction fine-tuning or alignment, raising the potential for the models to yield outputs that may be factually flawed.

# 7   Author Contributions

Takuya Akiba initiated the "Evolutionary Optimization of Model Merging Recipes" project, wrote the project design document, and initiated the parameter space model merging experiments, laying the groundwork for the methodology. Makoto Shing expanded the parameter space model merging to encompass vision-language models and diffusion models. Yujin Tang directed the efforts in data flow space model merging by incorporating ideas and methods inspired from the neural architecture search and morphology evolution literature, establishing foundational methods in this domain and in hybrid merging strategies. Qi Sun contributed to the implementation of our parameter space model merging framework and assisted in model evaluation. David Ha provided overarching guidance for the research project, offering technical insight, advice, feedback and writing.

# References

[1] Open AI. 2023. GPT-4V(ision) System Card. `https://cdn.openai.com/papers/GPTV_System_Card.pdf`

[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2623–2631. `https://doi.org/10.1145/3292500.3330701`

[3] augmxnt. 2023. *shisa-gamma-7b*. HuggingFace. `https://hf.co/augmxnt/shisa-gamma-7b-v1`

[4] AUTOMATIC1111. 2022. Stable Diffusion WebUI. `https://github.com/AUTOMATIC1111/stable-diffusion-webui`.

[5] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-VL: A Versatile Vision-Language Model for Understanding, Localization, Text Reading, and Beyond. arXiv:2308.12966 [cs.CV]

[6] Ethan Chern, Haoyang Zou, Xuefeng Li, Jiewen Hu, Kehua Feng, Junlong Li, and Pengfei Liu. 2023. Generative AI for Math: Abel. `https://github.com/GAIR-NLP/abel`.

[7] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems. *CoRR* abs/2110.14168 (2021). arXiv:2110.14168 `https://arxiv.org/abs/2110.14168`

[8] Nico Daheim, Thomas Möllenhoff, Edoardo Ponti, Iryna Gurevych, and Mohammad Emtiyaz Khan. 2024. Model Merging by Uncertainty-Based Gradient Matching. In *The Twelfth International Conference on Learning Representations*. `https://openreview.net/forum?id=D7KJmfEDQP`

[9] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. 2023. InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning. arXiv:2305.06500 [cs.CV]

[10] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation* 6, 2 (2002), 182–197.

[11] Gintare Karolina Dziugaite and Daniel M Roy. 2017. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008* (2017).

[12] Adam Gaier and David Ha. 2019. Weight agnostic neural networks. *Advances in neural information processing systems* 32 (2019).

[13] Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation. `https://doi.org/10.5281/zenodo.10256836`

[14] Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *arXiv preprint arXiv:2203.14680* (2022).

[15] Charles O. Goddard. 2024. mergekit. `https://github.com/arcee-ai/mergekit`

[16] David Ha, Andrew Dai, and Quoc V Le. 2016. Hypernetworks. *arXiv preprint arXiv:1609.09106* (2016).

[17] Nikolaus Hansen. 2006. The CMA evolution strategy: a comparing review. *Towards a new evolutionary computation: Advances in the estimation of distribution algorithms* (2006), 75–102.

[18] Sepp Hochreiter and Jürgen Schmidhuber. 1994. Simplifying neural nets by discovering flat minima. *Advances in neural information processing systems* 7 (1994).

[19] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Flat minima. *Neural computation* 9, 1 (1997), 1–42.

[20] HuggingFace. 2023. *Open LLM Leaderboard*. HuggingFace. `https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard`

[21] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089* (2022).

[22] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL]

[23] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. FastText.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651* (2016).

[24] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of Tricks for Efficient Text Classification. *arXiv preprint arXiv:1607.01759* (2016).

[25] Jean Kaddour, Linqing Liu, Ricardo Silva, and Matt J Kusner. 2022. When do flat minima optimizers work? *Advances in Neural Information Processing Systems* 35 (2022), 16577–16595.

[26] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2017. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. In *International Conference on Learning Representations*. https://openreview.net/forum?id=H1oyRlYgg

[27] Maxime Labonne. 2024. Automerger Experiment. *Tweet Thread* (2024). https://twitter.com/maximelabonne/status/1767124527551549860

[28] Maxime Labonne. 2024. Merge Large Language Models with mergekit. *Hugging Face Blog* (2024). https://huggingface.co/blog/mlabonne/merge-models

[29] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. arXiv:2301.12597 [cs.CV]

[30] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2023. Improved Baselines with Visual Instruction Tuning. arXiv:2310.03744 [cs.CV]

[31] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge. https://llava-vl.github.io/blog/2024-01-30-llava-next/

[32] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual Instruction Tuning. arXiv:2304.08485 [cs.CV]

[33] Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. WizardMath: Empowering Mathematical Reasoning for Large Language Models via Reinforced Evol-Instruct. *CoRR* abs/2308.09583 (2023). https://doi.org/10.48550/ARXIV.2308.09583 arXiv:2308.09583

[34] Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems* 35 (2022), 17703–17716.

[35] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems* 35 (2022), 17359–17372.

[36] nostalgebraist. 2021. Interpreting GPT: The Logit Lens. https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens. Accessed: 2024-03-08.

[37] Henning Petzka, Michael Kamp, Linara Adilova, Cristian Sminchisescu, and Mario Boley. 2021. Relative Flatness and Generalization. In *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (Eds.). https://openreview.net/forum?id=sygvo7ctb_

[38] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. 2019. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, Vol. 33. 4780–4789.

[39] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10684–10695.

[40] Jürgen Schmidhuber. 1992. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation* 4, 1 (1992), 131–139.

[41] Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. 2023. Language models are multilingual chain-of-thought reasoners. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. https://openreview.net/pdf?id=fR3wGCk-IXp

[42] Nobuyuki Shimizu, Na Rong, and Takashi Miyazaki. 2018. Visual Question Answering Dataset for Bilingual Image Understanding: A Study of Cross-Lingual Transfer Using Attention Maps. In *Proceedings of the 27th International Conference on Computational Linguistics* (Santa Fe, New Mexico, USA). Association for Computational Linguistics, 1918–1928. http://aclweb.org/anthology/C18-1163

[43] Makoto Shing and Takuya Akiba. 2023. Japanese Stable VLM. https://huggingface.co/stabilityai/japanese-stable-vlm

[44] David So, Quoc Le, and Chen Liang. 2019. The evolved transformer. In *International conference on machine learning*. PMLR, 5877–5886.

[45] Kenneth O Stanley and Risto Miikkulainen. 2002. Evolving neural networks through augmenting topologies. *Evolutionary computation* 10, 2 (2002), 99–127.

[46] Yujin Tang, Yingtao Tian, and David Ha. 2022. EvoJAX: Hardware-Accelerated Neuroevolution. *arXiv preprint arXiv:2202.05008* (2022).

[47] Tom White. 2016. Sampling generative networks. *arXiv preprint arXiv:1609.04468* (2016).

[48] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*. PMLR, 23965–23998.

[49] Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. 2023. TIES-Merging: Resolving Interference When Merging Models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). `http://papers.nips.cc/paper_files/paper/2023/hash/1644c9af28ab7916874f6fd6228a9bcf-Abstract-Conference.html`

[50] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language Models are Super Mario: Absorbing Abilities from Homologous Models as a Free Lunch. arXiv:2311.03099 [cs.CL]

[51] Barret Zoph and Quoc V Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578* (2016).

## A  Evaluation Details

In Table 1, GPT-3.5 is `gpt-3.5-turbo-0613`, GPT-4 is `gpt-4-0125-preview`, Llama 2 70B is `Llama-2-70b-chat`, Japanese Stable LM 70B is `japanese-stablelm-instruct-beta-70b` and Swallow 70B is `Swallow-70b-instruct-hf`.

For the evaluation using the Japanese LM Evaluation Harness, we utilized Stability AI Japan's fork of lm-eval-harness[3] [13] and configured it according to their convention. This configuration is widely used and compatible with the results on their report[4] and Rinna leaderboards[5], thus allowing direct comparison of scores with a large number of Japanese LLMs.

Table 4 is the full version of Table 2, allowing for comparisons between our models and a wider range of models. Some of the numbers are from the Stability AI Japan report and the Rinna leaderboard. Despite having only 7B to 10B parameters, our models achieve higher scores than all existing <70B parameters models and exhibit scores on par with the SOTA 70B models.

## B  Evolving for License Specific Open-Source Models

In the main section, our `EvoLLM-JP` results were evolved using models found on HuggingFace. However, some of the models uses, in particular, `WizardMath-7B-V1.1` [33] has been released under a Non-Commercial, Research-only, Microsoft License, which is not *truly* open-source. Therefore, our release of `EvoLLM-JP` is also released under a Non-Commercial, Research-only License to be consistent with the `WizardMath-7B-V1.1` model.

As researchers who benefited from the open-source community, we would like for models that we release to also be under an open-source license. In the spirit of open-source, and to showcase the applicability of our method to tackle even challenging issues like model licenses. We have ran a similar experiment where we incorporated *only* models that have been released under a true open-source license, such as MIT or Apache 2.0, and have produced a similar performing model called `EvoLLM-JP-A`, which we will release under Apache 2.0.[6]

Specifically, our `EvoLLM-JP-A` is a merge of shisa-gamma-7b-v1, Arithmo2-Mistral-7B, and Abel-7B-002, all of which are under MIT or Apache 2.0 License. The MGSM-JA score measured using the protocol described in Section 4.1 is 52.4, and the Japanese Language Model Evaluation Harness score is 69.0. We have included results of this Apache 2.0-licensed model for comparison in Table 4, which provides a more comprehensive comparison than Table 2 in the main text.

## C  Case Study

Table 5 provides an example of responses to a mathematical question by existing models and our model. By merging a Japanese language model, we not only improve our capability to understand and use Japanese in reading and writing but also expand our knowledge about Japan. This example requires both mathematical reasoning and Japanese-specific knowledge, specifically that Setsubun is the day before the beginning of spring (Risshun). Notably, only our merged model provides the correct answer. Even when the question is translated into English, the English math model `WizardMath-7B-V1.1` fails to answer correctly, demonstrating that our merged model is superior to a combination of machine translation and an English math model.

It is worth noting that in Table 5, we also provide the answer from the general-purpose math model `WizardMath-7B-V1.1` with the question translated into English. However, even with the translated question, `WizardMath-7B-V1.1` fails to provide the correct answer. This highlights the fact that simply translating the question is not sufficient to solve the problem, as it also requires an understanding of the Japanese cultural context. Even if one were to consider a combination of machine translation and an English math model as an alternative approach, it would still fall short in capturing the nuances and context-specific knowledge required to answer the question correctly.

---

[3] `https://github.com/Stability-AI/lm-evaluation-harness/tree/jp-stable`
[4] `https://ja.stability.ai/blog/japanese-stable-lm-beta`
[5] `https://rinnakk.github.io/research/benchmarks/lm/index.html`
[6] This model will also be released on our GitHub: `https://github.com/SakanaAI/`

This demonstrates the unique value of our merged model, which effectively combines mathematical reasoning capabilities with Japanese language understanding and cultural knowledge.

Additionally, Table 6 provides the case studies for our VLM. We observe that our VLM successfully obtained knowledge of the Japanese LLM. In the 1st example, our VLM generates the correct answer (Koi-nobori) while the original VLM (`LLaVA-1.6-Mistral-7B`) answers incorrectly. Compared to the Japanese VLM (JSVLM), our VLM generally describes the photo more precisely as the 2nd example of a deer. Also, we found that the original VLM hallucinates a lot in Japanese such that it answers the color of traffic light is green while our VLM answers correctly.

Table 4: **Breakdown of JP-LMEH Scores for Japanese Language Proficiency (Full Version of Table 2)**. JP-LMEH (Japanese Language Model Evaluation Harness) is a benchmark suite consisting of 9 tasks, and the average score (Avg column) is used as an indicator of overall Japanese language proficiency.

| Model | Size | JComQA | JNLI | MARC | JSQuAD | JAQKET | XLSum | XWino | MGSM | JCoLA | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|
| shisa-gamma-7b-v1 | 7B | 91.2 | 72.1 | 94.6 | 73.9 | 68.0 | 25.9 | 80.5 | 29.6 | 58.7 | 66.1 |
| WizardMath-7B-V1.1 | 7B | 74.7 | 42.7 | 90.4 | 84.6 | 68.5 | 22.3 | 69.8 | 38.8 | 48.9 | 60.1 |
| Abel-7B-002 | 7B | 70.3 | 51.8 | 62.3 | 83.8 | 69.0 | 22.5 | 68.2 | 28.0 | 52.7 | 56.5 |
| **Ours (PS)** | 7B | 89.1 | 65.7 | 95.4 | 89.5 | 77.7 | 25.5 | 81.2 | 50.0 | 60.5 | **70.5** |
| **Ours (DFS)** | 10B | 67.7 | 58.2 | 53.5 | 66.8 | 54.3 | 17.3 | 65.6 | 30.0 | 65.6 | **53.2** |
| **Ours (PS+DFS)** | 10B | 88.2 | 50.3 | 91.5 | 78.6 | 77.8 | 23.2 | 73.0 | 40.0 | 73.0 | **66.2** |
| **Ours (PS-A; see Appendix B)** | 7B | 87.0 | 62.3 | 91.9 | 90.4 | 77.4 | 23.0 | 78.9 | 46.4 | 63.6 | **69.0** |
| Swallow-70b-instruct-hf | 70B | 95.3 | 57.2 | 91.7 | 94.1 | 93.9 | 23.1 | 83.3 | 45.2 | 59.5 | 71.5 |
| Swallow-70b-hf | 70B | 94.8 | 55.8 | 87.8 | 93.5 | 93.8 | 23.2 | 83.7 | 47.2 | 60.7 | 71.2 |
| japanese-stablelm-base-beta-70b | 70B | 93.7 | 45.2 | 91.0 | 94.2 | 92.8 | 25.4 | 83.6 | 41.2 | 63.8 | 70.1 |
| nekomata-14b-instruction | 14B | 93.7 | 57.5 | 90.0 | 93.6 | 89.0 | 25.2 | 77.4 | 35.2 | 67.3 | 69.9 |
| japanese-stablelm-instruct-beta-70b | 70B | 91.2 | 50.4 | 92.9 | 87.1 | 88.4 | 24.3 | 82.0 | 37.2 | 61.7 | 68.3 |
| nekomata-14b | 14B | 93.1 | 41.7 | 84.1 | 94.2 | 90.8 | 21.3 | 78.6 | 37.6 | 65.0 | 67.4 |
| youri-7b-chat | 7B | 91.8 | 70.3 | 96.7 | 79.6 | 83.7 | 24.2 | 80.9 | 25.2 | 53.8 | 67.4 |
| Llama-2-70b-hf | 70B | 89.3 | 53.0 | 70.4 | 93.2 | 89.5 | 23.8 | 79.8 | 40.0 | 65.4 | 67.2 |
| nekomata-7b-instruction | 7B | 88.0 | 74.7 | 92.3 | 91.3 | 82.5 | 21.1 | 76.8 | 19.2 | 53.8 | 66.6 |
| youri-7b-instruction | 7B | 88.8 | 63.6 | 93.8 | 92.2 | 83.9 | 24.7 | 78.9 | 17.2 | 54.0 | 66.3 |
| Qwen-14B | 14B | 89.5 | 65.2 | 76.7 | 93.0 | 79.5 | 18.4 | 71.6 | 37.2 | 61.0 | 65.8 |
| Swallow-MX-8x7b-NVE-v0.1 | 47B | 92.0 | 52.4 | 59.8 | 93.1 | 91.9 | 26.1 | 72.9 | 44.8 | 59.2 | 65.8 |
| youri-7b-chat-gptq | 7B | 89.5 | 68.0 | 96.0 | 88.6 | 79.7 | 15.6 | 79.1 | 21.6 | 53.8 | 65.8 |
| nekomata-7b-instruction-gguf | 7B | 87.9 | 68.4 | 89.6 | 89.9 | 79.8 | 20.8 | 77.3 | 19.2 | 55.3 | 65.3 |
| nekomata-14b-instruction-gguf | 14B | 93.3 | 57.8 | 51.7 | 91.3 | 86.2 | 23.6 | 77.4 | 35.2 | 68.7 | 65.0 |
| Llama-2-70b-chat-hf | 70B | 80.2 | 53.4 | 94.4 | 91.6 | 80.1 | 21.8 | 73.6 | 30.4 | 54.6 | 64.5 |
| youri-7b-instruction-gptq | 7B | 88.1 | 62.0 | 93.5 | 90.6 | 74.4 | 15.3 | 78.2 | 15.2 | 53.3 | 63.4 |
| japanese-stablelm-base-gamma-7b | 7B | 89.5 | 37.6 | 79.2 | 90.6 | 89.5 | 22.9 | 82.9 | 20.8 | 55.6 | 63.2 |
| Swallow-13b-instruct-hf | 13B | 89.0 | 43.1 | 74.1 | 92.8 | 92.5 | 20.6 | 70.4 | 24.0 | 61.9 | 63.2 |
| japanese-stablelm-instruct-gamma-7b | 7B | 91.4 | 40.7 | 72.9 | 89.9 | 88.2 | 22.8 | 82.2 | 17.2 | 56.4 | 62.4 |
| Swallow-13b-hf | 13B | 86.0 | 44.8 | 74.5 | 92.2 | 92.2 | 21.6 | 71.4 | 19.2 | 59.5 | 62.4 |
| nekomata-14b-gguf | 14B | 93.0 | 41.1 | 50.0 | 93.0 | 88.3 | 22.2 | 80.5 | 32.4 | 60.2 | 62.3 |
| Swallow-MS-7b-v0.1 | 7B | 88.3 | 35.8 | 83.1 | 90.3 | 89.4 | 20.7 | 74.6 | 22.4 | 55.5 | 62.2 |
| Swallow-7b-instruct-hf | 7B | 86.8 | 36.3 | 94.2 | 90.0 | 89.4 | 17.4 | 76.8 | 10.4 | 50.0 | 61.2 |
| llm-jp-13b-instruct-full-jaster-dolly-oasst-v1.0 | 13B | 83.2 | 74.5 | 74.9 | 94.0 | 85.1 | 9.6 | 71.3 | 3.2 | 50.4 | 60.7 |
| Swallow-7b-hf | 7B | 80.2 | 37.9 | 93.1 | 89.2 | 90.1 | 18.3 | 76.6 | 10.4 | 50.0 | 60.7 |
| nekomata-7b | 7B | 82.9 | 38.6 | 67.0 | 89.8 | 82.5 | 16.9 | 74.5 | 16.0 | 60.0 | 58.7 |
| japanese-stablelm-instruct-beta-7b | 7B | 81.9 | 39.9 | 88.3 | 84.1 | 78.2 | 20.8 | 75.7 | 4.8 | 51.7 | 58.4 |
| japanese-stablelm-base-beta-7b | 7B | 76.8 | 38.0 | 86.0 | 87.1 | 79.5 | 21.9 | 76.6 | 7.6 | 49.2 | 58.1 |
| Qwen-7B | 7B | 76.0 | 46.9 | 83.3 | 89.7 | 69.2 | 13.8 | 60.6 | 22.0 | 57.3 | 57.6 |
| youri-7b-gptq | 7B | 78.9 | 40.5 | 72.9 | 85.7 | 78.1 | 16.3 | 78.3 | 4.8 | 54.0 | 57.2 |
| youri-7b | 7B | 82.8 | 35.9 | 90.1 | 57.2 | 82.5 | 20.5 | 78.0 | 6.8 | 58.5 | 56.9 |
| ELYZA-japanese-Llama-2-7b-instruct | 7B | 79.2 | 38.8 | 89.9 | 83.2 | 68.7 | 16.8 | 71.1 | 11.2 | 50.5 | 56.6 |
| weblab-10b-instruction-sft | 10B | 63.6 | 42.7 | 86.4 | 87.8 | 73.9 | 23.1 | 72.4 | 2.8 | 50.5 | 55.9 |
| Mistral-7B-v0.1 | 7B | 74.0 | 44.2 | 63.3 | 85.7 | 69.8 | 22.8 | 69.9 | 20.4 | 51.1 | 55.7 |
| ELYZA-japanese-Llama-2-7b | 7B | 75.2 | 36.1 | 87.3 | 86.4 | 68.0 | 16.3 | 72.0 | 5.2 | 53.1 | 55.5 |
| nekomata-7b-gguf | 7B | 82.3 | 33.7 | 50.0 | 86.8 | 74.9 | 19.9 | 78.3 | 13.6 | 55.8 | 55.0 |
| japanese-stablelm-instruct-ja_vocab-beta-7b | 7B | 79.9 | 39.2 | 79.5 | 82.0 | 78.9 | 9.9 | 68.9 | 6.0 | 49.2 | 54.8 |
| japanese-stablelm-base-ja_vocab-beta-7b | 7B | 69.9 | 38.8 | 80.0 | 83.8 | 80.1 | 7.5 | 72.5 | 3.6 | 50.0 | 54.0 |
| calm2-7b-chat | 7B | 70.3 | 33.7 | 87.7 | 82.6 | 77.8 | 5.4 | 70.9 | 5.6 | 51.9 | 54.0 |
| stockmark-13b | 13B | 70.7 | 36.2 | 56.5 | 87.2 | 87.3 | 7.7 | 73.4 | 9.6 | 50.0 | 53.2 |
| Llama-2-7b-hf | 7B | 65.9 | 35.1 | 83.2 | 82.7 | 70.6 | 19.0 | 65.2 | 6.8 | 47.1 | 52.9 |
| calm2-7b | 7B | 58.2 | 33.3 | 81.2 | 81.8 | 81.9 | 2.4 | 73.0 | 5.6 | 53.3 | 52.3 |
| plamo-13b | 13B | 54.5 | 33.8 | 85.0 | 78.6 | 74.5 | 12.9 | 70.5 | 2.4 | 54.5 | 51.9 |
| weblab-10b | 10B | 44.0 | 30.7 | 64.0 | 83.3 | 74.6 | 20.3 | 71.4 | 2.0 | 50.8 | 49.0 |
| llm-jp-13b-v1.0 | 13B | 39.6 | 35.0 | 67.4 | 79.3 | 78.0 | 10.9 | 69.5 | 2.0 | 54.4 | 48.5 |
| plamo-13b-instruct-nc | 13B | 45.8 | 35.3 | 91.6 | 75.2 | 63.5 | 5.8 | 65.2 | 1.2 | 50.6 | 48.3 |
| plamo-13b-instruct | 13B | 42.5 | 34.9 | 90.0 | 77.3 | 67.0 | 5.4 | 62.9 | 2.0 | 50.0 | 48.0 |

Table 5: **Case Study of EvoLLM-JP-v1-7B**. This study presents the responses of the LLMs to a problem that demands both Japanese-specific knowledge and mathematical reasoning.

| | *Problem* |
|---|---|
| **JA Instruction** | 去年の立春は2月3日でした。今年の立春は2月4日です。去年の節分の月、去年の節分の日、今年の節分の月、今年の節分の日を足し合わせると、いくつになりますか？ |
| **EN Instruction** | Last year, Risshun was on February 3rd. This year, it is on February 4th. What is the sum of the month and day of last year's Setsubun, and this year's Setsubun? |
| **Ground Truth Answer** | 9 |
| **Note** | Setsubun, the eve of the first day of spring, always falls on the day before Risshun, which marks the start of spring according to the lunar calendar. The date of Risshun can vary annually. |

| | *Answers* |
|---|---|
| **EvoLLM-JP-v1-7B** JA Instruction | 去年の立春は2月3日で、今年の立春は2月4日です。去年の節分は立春の前日、つまり2月2日でした。今年の節分は立春の前日、つまり2月3日です。去年の節分の月は2月、去年の節分の日は2日で、合計は2 + 2 = 4です。今年の節分の月は2月、今年の節分の日は3日で、合計は2 + 3 = 5です。足し合わせると、4 + 5 = 9になります。 ####9 The answer is: 9. |
| **shisa-gamma-7b** JA Instruction | まず、去年の立春の日付を計算してみましょう。 去年の立春の日付 = 2021年の2月3日 次に、去年の節分の日付を計算してみましょう。 去年の節分の日付 = 2021年の2月2日 次に、今年の立春の日付を計算してみましょう。 今年の立春の日付 = 2022年の2月4日 次に、今年の節分の日付を計算してみましょう。 今年の節分の日付 = 2022年の2月1日 最後に、去年の節分の月と日、今年の節分の月と日を足し合わせてみましょう。 去年の節分の月と日 = 2021年の2月2日 今年の節分の月と日 = 2022年の2月1日 これで、足し合わせることができます。 足し合わせた月と日 = 2021年の2月2日 + 2022年の2月1日 = 2023年の2月3日 つまり、去年の節分の月と日、今年の節分の月と日を足し合わせると、2023年の2月3日になります。 |
| **WizardMath-7B-V1.1** JA Instruction | Step 1: Identify the dates for the given events. - Last year's Setsubun (February 3) - This year's Setsubun (February 4) Step 2: Determine the dates for the corresponding events in the previous and next years. - Last year's Setsubun (February 3) falls on a Monday. - This year's Setsubun (February 4) falls on a Tuesday. Step 3: Calculate the number of days between the corresponding events. - The number of days between last year's and this year's Setsubun is 1 day (Tuesday - Monday = 1 day). Step 4: Determine the dates for the corresponding events in the lunar calendar. - Last year's Setsubun was on the 1st day of the 2nd month. - This year's Setsubun will be on the 2nd day of the 2nd month. Step 5: Calculate the number of days between the corresponding events in the lunar calendar. - The number of days between last year's and this year's Setsubun in the lunar calendar is 1 day (2nd day - 1st day = 1 day). Step 6: Add the number of days between the corresponding events in both calendars. - The total number of days between last year's and this year's Setsubun is 1 (from the solar calendar) + 1 (from the lunar calendar) = 2 days. Step 7: Write the final answer. - The sum of the dates for the corresponding events is 2. The answer is: 2. |
| **WizardMath-7B-V1.1** EN Instruction | We are given that last year, Risshun (the beginning of spring according to the lunisolar Japanese calendar) was on February 3rd, and this year it is on February 4th. We need to find the sum of the month and day of last year's Setsubun, and this year's Setsubun. Step 2: Identify the month and day of last year's Setsubun. Last year, Risshun was on February 3rd. So, the month is February (2) and the day is 3. Step 3: Identify the month and day of this year's Setsubun. This year, Risshun is on February 4th. So, the month is February (2) and the day is 4. Step 4: Calculate the sum of the month and day of last year's and this year's Setsubun. The sum of the month and day of last year's Setsubun is 2 + 3 = 5. The sum of the month and day of this year's Setsubun is 2 + 4 = 6. Step 5: Add the two sums together. The total sum of the month and day of last year's and this year's Setsubun is 5 + 6 = 11. So, the sum of the month and day of last year's Setsubun, and this year's Setsubun is 11. The answer is: 11. |

Table 6: **Case Study of EvoVLM-JP**. This study presents the responses of the VLMs to questions that demand both Japanese-specific knowledge and VQA abilities.

| | *Example 1* |
|---|---|



| User | この画像に映っている魚の形をした飾りは、日本では何と呼ばれていますか？<br>(What is the fish-shaped decoration in this image called in Japan?) |
|---|---|
| **LLaVA-1.6-Mistral-7B** | この画像に映っている魚の形をした飾りは、日本では「花魚」と呼ばれています。<br>(The fish-shaped decoration in this image is called "hana-gyo" in Japan.) |
| **Japanese Stable VLM** | こいのぼり (Koi-nobori) |
| **EvoVLM-JP** | この画像に映っている魚の形をした飾りは、日本では「鯉のぼり」と呼ばれています。<br>(The fish-shaped decoration in this image is called "Koi-nobori" in Japan.) |

| | *Example 2* |
|---|---|



| User | 写真を説明してください。<br>(Please explain this photo.) |
|---|---|
| **LLaVA-1.6-Mistral-7B** | この写真では、自然の中で小さな動物が歩いています。その動物は、犬のような外貌を持っていますが、実際には犬ではありません。この動物は、野生動物として知られていますが、具体的な種類はわかりません。背景には、秋の季節になっていることがわかります。多くの樹木には、黄色い葉が落ちています。このような環境で、動物が歩いていることを見ることができます。この写真では、動物が自然の中で活動していることを表現しています。同時に、秋の季節になっていることを伝えることもできます。<br>(In this photo, a small animal is walking in nature. The animal has a canine-like appearance but is not actually a dog. This creature is known as a wild animal, but the specific species is unknown. In the background, it is evident that it is autumn. Many trees have yellow leaves falling. In such an environment, the animal can be seen walking. The photo depicts the animal being active in nature while also conveying that it is autumn.) |
| **Japanese Stable VLM** | 紅葉と鹿の写真です。<br>(It's a photo of autumn leaves and deer.) |
| **EvoVLM-JP** | この写真は、一匹の小さな鹿が落ち葉の上に立っている様子を捉えています。鹿は警戒しているようで、周囲を注意深く見回しています。背景には紅葉した木々が見え、秋の雰囲気が感じられます。<br>(This photo captures a small deer standing on fallen leaves. The deer appears to be alert, carefully scanning its surroundings. In the background, you can see trees with autumn foliage, creating an atmosphere of autumn.) |

*Example 3*



| User | この信号機の色は何色ですか?<br>(What color is this traffic light?) |
|---|---|
| **LLaVA-1.6-Mistral-7B** | (A) この信号機の色は緑色です。 (The color of this traffic light is green.)<br>(B) この信号機の色は緑色です。 (The color of this traffic light is green.)<br>(C) この信号機の色は緑色です。 (The color of this traffic light is green.) |
| **Japanese Stable VLM** | (A) 緑色です。 (It is green.)<br>(B) 緑と白です。 (It is green and white.)<br>(C) 白色 (White) |
| **EvoVLM-JP** | (A) この信号機の色は青です。 (The color of this traffic light is blue.)<br>(B) この信号機の色は青です。 (The color of this traffic light is blue.)<br>(C) この信号機の色は青です。 (The color of this traffic light is blue.) |

*Example 4*
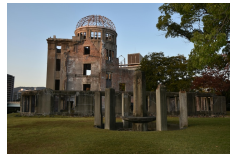


| User | この建物について教えてください。<br>(Please explain this building.) |
|---|---|
| **LLaVA-1.6-Mistral-7B** | この建物は、日本で位置しています。具体的には、東京都渋谷区にあります。この建物は、1945年8月15日に、第二次世界大戦中に日本軍によって攻撃され、完全に破壊されました。その後、1964年に再構築されました。この建物は、「平和の塔」（Peace Tower）と呼ばれています。その名前は、この建物が再構築される際に決められた名前です。この建物は、平和を祈るために作られたものであり、国内外から多くの人々が訪れる場所です。<br>(This building is located in Japan, specifically in Shibuya Ward, Tokyo. It was completely destroyed by the Japanese military during World War II on August 15, 1945. Subsequently, it was reconstructed in 1964. This building is called the "Peace Tower," a name decided upon during its reconstruction. It was created as a symbol of peace, and it is a place visited by many people from both within and outside the country to pray for peace.) |
| **Japanese Stable VLM** | 原爆ドーム<br>(Atomic Bomb Dome) |
| **EvoVLM-JP** | この建物は、日本の広島市にある「原爆ドーム」です。 1945年8月6日、広島市に原子爆弾が投下されました。 この原子爆弾は、広島市の中心部にある「広島県産業奨励館」に直撃しました。 この奨励館は、現在は「原爆ドーム」と呼ばれています。 この原爆ドームは、原子爆弾の被害を受けた建物の中で、唯一残っている建物です。 この原爆ドームは、1996年に世界遺産に登録されました。<br>(This building is the "Atomic Bomb Dome" located in Hiroshima City, Japan. On August 6, 1945, an atomic bomb was dropped on Hiroshima City. The bomb directly hit the "Hiroshima Prefectural Industrial Promotion Hall," located in the central part of Hiroshima City. This promotion hall is now known as the "Atomic Bomb Dome." It is the only building that remains standing among those damaged by the atomic bomb. The Atomic Bomb Dome was registered as a UNESCO World Heritage Site in 1996.) |