Step 1: Modular Decomposition

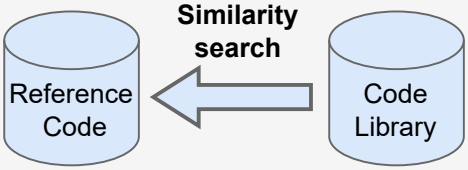
I have a natural language description of a computational task. Can you help me decompose it into smaller, more concise task modules? The description is:

> I need to design a workflow to select the optimal image classification model for images. I want to apply the ResNet, ViT, and DenseNet models respectively. During the model training phase, I will use the .....

Chain-of-Thought

Step 2: Code Generation

I have a concise task module, can you help me generate code for it? I can give you some reference code:



The task is:

> Model Training Module:
> Train and validate each applied model using the same training data and validation data.

Code_n:

```
def train_model(ResNet):
    command = f"python train.py --model {model_name} --train_data <train_data> --validation_data <validation_data>"
    return couler.run_container(
        image="training-image",
        command=command.split(" "),      )
```

Step 3: Self-calibration

I have generated some code for a specific task. Can you evaluate it and provide a score between 0 and 1 to indicate its compliance with predefined templates and standards? A score of 1 means the code fully complies with the standards and templates, and a score of 0 means it does not comply at all. The generated code is: [code_n]

Subtask in Chains

Step 4: User Feedback

When users execute the generated workflow code, they can submit feedback or suggest modifications to the Large Language Model (LLM) should they encounter any bugs during the execution.