(mm-processing)=

Multi-Modal Data Processing

To enable various optimizations in vLLM such as [chunked prefill](#chunked-prefill) and [prefix caching](#automatic-prefix-caching), we use {class}`~vllm.multimodal.processing.BaseMultiModalProcessor` to provide the correspondence between placeholder feature tokens (e.g. `<image>`) and multi-modal inputs (e.g. the raw input image) based on the outputs of HF processor.

Here are the main features of {class}`~vllm.multimodal.processing.BaseMultiModalProcessor`:

Prompt Replacement Detection

One of the main responsibilies of HF processor is to replace input placeholder tokens (e.g. `<image>` for a single image) with feature placeholder tokens (e.g. `<image><image>...<image>`, the number of which equals to the feature size). The information about which tokens have been replaced is key to finding the correspondence between placeholder feature tokens and multi-modal inputs.

In vLLM, this information is specified using {class}`~vllm.multimodal.processing.PromptReplacement` in {meth}`~vllm.multimodal.processing.BaseMultiModalProcessor._get_prompt_replacements`. Given this specification, we can automatically detect whether HF has replaced the input placeholder tokens by checking whether the feature placeholder tokens exist in the prompt.

Tokenized Prompt Inputs

To enable tokenization in a separate process, we support passing input token IDs alongside multi-modal data.

The problem

Consider that HF processors follow these main steps:

1. Tokenize the text
2. Process multi-modal inputs
3. Perform prompt replacement

And we require that:

- For text + multi-modal inputs, apply all steps 1--3.
- For tokenized + multi-modal inputs, apply only steps 2--3.

How can we achieve this without rewriting HF processors? We can try to call the HF processor several times on different inputs:

- For text + multi-modal inputs, simply call the HF processor directly.
- For tokenized + multi-modal inputs, call the processor only on the multi-modal inputs.

While HF processors support text + multi-modal inputs natively, this is not so for tokenized + multi-modal inputs: an error is thrown if the number of input placeholder tokens do not correspond to the number of multi-modal inputs.

Moreover, since the tokenized text has not passed through the HF processor, we have to apply Step 3 by ourselves to keep the output tokens and multi-modal data consistent with each other.

(mm-dummy-text)=

Dummy text

We work around the first issue by requiring each model to define how to generate dummy text based on the number of multi-modal inputs, via {meth}`~vllm.multimodal.profiling.BaseDummyInputsBuilder.get_dummy_processor_inputs`. This lets us generate dummy text corresponding to the multi-modal inputs and input them together to obtain the processed multi-modal data.

(mm-automatic-prompt-replacement)=

Automatic prompt replacement

We address the second issue by implementing model-agnostic code in {meth}`~vllm.multimodal.processing.BaseMultiModalProcessor._apply_prompt_replacements` to automatically replace input placeholder tokens with feature placeholder tokens based on the specification outputted by {meth}`~vllm.multimodal.processing.BaseMultiModalProcessor._get_prompt_replacements`.

Summary

With the help of dummy text and automatic prompt replacement, our multi-modal processor can finally accept both text and token prompts with multi-modal data. The detailed logic is shown in {meth}`~vllm.multimodal.processing.BaseMultiModalProcessor._apply_hf_processor_main`.

Processor Output Caching

Some HF processors, such as the one for Qwen2-VL, are [very slow](gh-issue:9238). To alleviate this problem, we cache the multi-modal outputs of HF processor to avoid processing the same multi-modal input (e.g. image) again.

When new data is passed in, we first check which items are in the cache, and which ones are missing. The missing items are passed into the HF processor in a single batch and cached, before being merged with the existing items in the cache.

Since we only process the missing multi-modal data items, the number of input placeholder tokens no longer corresponds to the number of the multi-modal inputs, so they can't be passed alongside the text prompt to HF processor. Therefore, we process the text and multi-modal inputs separately, using [dummy text](#mm-dummy-text) to avoid HF errors. Since this skips HF's prompt replacement code, we apply [automatic prompt replacement](#mm-automatic-prompt-replacement) afterwards to keep the output tokens and multi-modal data consistent with each other.