

```
{ "added_tokens_decoder": { "128000": { "content": "<|begin_of_text|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128001": { "content": "<|end_of_text|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128002": { "content": "<|im_start|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128003": { "content": "<|im_end|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128004": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128005": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128006": { "content": "<|start_header_id|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128007": { "content": "<|end_header_id|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128008": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128009": { "content": "<|eot_id|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128010": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128011": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128012": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128013": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128014": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128015": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128016": { "content": "", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": false }, "128017": { "content": "<|reserved_special_token_12|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128018": { "content": "<|reserved_special_token_13|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128019": { "content": "<|reserved_special_token_14|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128020": { "content": "<|reserved_special_token_15|>", "lstrip": false, "normalized": false, "rstrip": false, "single_word": false, "special": true }, "128021": {
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```
"content": "<|reserved_special_token_250|>", "lstrip": false, "normalized": false, "rstrip": false,
"single_word": false, "special": true } }, "bos_token": "<|begin_of_text|>", "chat_template": [ { "name":
"default", "template": "{{bos_token}}{% for message in messages %}{{<|im_start|>' + message['role']
+ '\n' + message['content'] + '<|im_end|>' + '\n'}}{% endfor %}}{% if add_generation_prompt %}{{
'<|im_start|>assistant\n' }}{% endif %}" }, { "name": "tool_use", "template": "{%- macro
json_to_python_type(json_spec) %}\n{%- set basic_type_map = {\n \"string\": \"str\", \n \"number\":
\"float\", \n \"integer\": \"int\", \n \"boolean\": \"bool\" \n} %}\n\n{%- if basic_type_map[json_spec.type] is
defined %}\n {{- basic_type_map[json_spec.type] }}\n{%- elif json_spec.type == \"array\" %}\n {{-
\"list[\" + json_to_python_type(json_spec.items) + \"]\" }}\n{%- elif json_spec.type == \"object\" %}\n
{%- if json_spec.additionalProperties is defined %}\n {{- \"dict[str, \" +
json_to_python_type(json_spec.additionalProperties) + \"]\" }}\n {%- else %}\n {{- \"dict\" }}\n {%- endif
%}\n{%- elif json_spec.type is iterable %}\n {{- \"Union[\" }}\n {%- for t in json_spec.type %}\n {{-
json_to_python_type({\"type\": t}) }}\n {%- if not loop.last %}\n {{- \",\" }} \n {%- endif %}\n {%- endfor
%}\n {{- \"]\" }}\n{%- else %}\n {{- \"Any\" }}\n{%- endif %}\n{%- endmacro %}\n\n{{- bos_token
}}\n{{- '<|im_start|>system\n' }}\n{{- \"You are a function calling AI model. You are provided with
function signatures within XML tags. You may call one or more functions to assist with the user
query. Don't make assumptions about what values to plug into functions. Here are the available
tools: \" }}\n{%- for tool in tools %}\n {%- if tool.function is defined %}\n {%- set tool = tool.function
%}\n {%- endif %}\n {{- '{\"type\": \"function\", \"function\": ' }}\n {{- '{\"name\": \"\" + tool.name + '\", ' }}\n
{{- \"description\": \"\" + tool.name + '(' }}\n {%- for param_name, param_fields in
tool.parameters.properties.items %}\n {{- param_name + \": \" + json_to_python_type(param_fields)
}}\n {%- if not loop.last %}\n {{- \", \" }}\n {%- endif %}\n {%- endfor %}\n {{- \")\" }}\n {%- if tool.return
is defined %}\n {{- \" -> \" + json_to_python_type(tool.return) }}\n {%- endif %}\n {{- \" - \" +
tool.description + \"\n\n\" }}\n {%- for param_name, param_fields in tool.parameters.properties.items
%}\n {%- if loop.first %}\n {{- \" Args:\n\" }}\n {%- endif %}\n {{- \" \" + param_name + \"(\" +
json_to_python_type(param_fields) + \"): \" + param_fields.description|trim }}\n {%- endfor %}\n {%-
if tool.return is defined and tool.return.description is defined %}\n {{- \"\n Returns:\n \" +
```

```

tool.return.description }}\n {% - endif %}\n {{- \"\" }}\n {{- ', \"parameters\": ' }}\n {% - if
tool.parameters.properties | length == 0 %}\n {{- \"{}\" }}\n {% - else %}\n {{- tool.parameters|tojson
}}\n {% - endif %}\n {{- \"\n\" }}\n {% - if not loop.last %}\n {{- \"\n\" }}\n {% - endif %}\n{% - endfor %}\n{{-
\" \" }}\n{{- 'Use the following pydantic model json schema for each tool call you will make:
{\"properties\": {\"name\": {\"title\": \"Name\", \"type\": \"string\"}, \"arguments\": {\"title\":
\"Arguments\", \"type\": \"object\"}}, \"required\": [\"name\", \"arguments\"], \"title\": \"FunctionCall\",
\"type\": \"object\"}}\n' }}\n{{- \"For each function call return a json object with function name and
arguments within XML tags as follows:\n\" }}\n{{- \"\n\" }}\n{{- '{\"name\": , \"arguments\": }' }}\n{{-
'<|im_end|>\n' }}\n{% - for message in messages %}\n {% - if message.role == \"user\" or
message.role == \"system\" or (message.role == \"assistant\" and message.tool_calls is not defined)
%}\n {{- '<|im_start|>' + message.role + '\n' + message.content + '<|im_end|>' + '\n' }}\n {% - elif
message.role == \"assistant\" %}\n {{- '<|im_start|>' + message.role }}\n {% - for tool_call in
message.tool_calls %}\n {{- '\n\n' }} {% - if tool_call.function is defined %}\n {% - set tool_call =
tool_call.function %}\n {% - endif %}\n {{- '{' }}\n {{- '\"name\": ' }}\n {{- tool_call.name }}\n {{- '\"' }}\n
{{- ', ' }}\n {% - if tool_call.arguments is defined %}\n {{- '\"arguments\": ' }}\n {% - if tool_call.arguments
is string %}\n {{- tool_call.arguments }}\n {% - else %}\n {{- tool_call.arguments|tojson }}\n {% - endif
%}\n {% - endif %}\n {{- '}' }}\n {{- '\n' }}\n {% - endfor %}\n {{- '<|im_end|>\n' }}\n {% - elif message.role
== \"tool\" %}\n {% - if loop.previtem and loop.previtem.role != \"tool\" %}\n {{- '<|im_start|>tool\n' }}\n
{% - endif %}\n {{- '\n' }}\n {{- message.content }}\n {% - if not loop.last %}\n {{- '\n\n' }}\n {% - else %}\n
{{- '\n' }}\n {% - endif %}\n {% - if not loop.last and loop.nextitem.role != \"tool\" %}\n {{- '<|im_end|>'
}}\n {% - elif loop.last %}\n {{- '<|im_end|>' }}\n {% - endif %}\n {% - endif %}\n{% - endfor %}\n{% - if
add_generation_prompt %}\n {{- '<|im_start|>assistant\n' }}\n{% - endif %}\n\" } ],
\"clean_up_tokenization_spaces\": true, \"eos_token\": \"<|im_end|>\", \"model_input_names\": [
\"input_ids\", \"attention_mask\" ], \"model_max_length\": 1000000000000000019884624838656,
\"pad_token\": \"<|end_of_text|>\", \"tokenizer_class\": \"PreTrainedTokenizerFast\" }

```