**OPEA Service Spec**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| 0.9 | 26-Aug-2012 | [feng.tian@intel.com](mailto:feng.tian@intel.com)  [liang1.lv@intel.com](mailto:liang1.lv@intel.com)  [haihao.shen@intel.com](mailto:haihao.shen@intel.com)  kaokao.lv@intel.com | Initial draft |

Index

Contents

[OPEA Mega Service API 4](#_Toc170623328)

[1. List Services 4](#_Toc170623329)

[Request 4](#_Toc170623330)

[Response 4](#_Toc170623331)

[2. List Configurable Parameters 4](#_Toc170623332)

[Request 4](#_Toc170623333)

[Response 5](#_Toc170623334)

[3. Embedding 5](#_Toc170623335)

[Request 5](#_Toc170623336)

[Response 6](#_Toc170623337)

[4. Chat 6](#_Toc170623338)

[Request 6](#_Toc170623339)

[Response 7](#_Toc170623340)

[5. Other operations 8](#_Toc170623341)

[OPEA Micro Service API 9](#_Toc170623342)

[1. Embedding Micro Service 9](#_Toc170623343)

[Request 9](#_Toc170623344)

[Response 9](#_Toc170623345)

[2. LLM Generation Micro Service 9](#_Toc170623346)

[Request 10](#_Toc170623347)

[Response 10](#_Toc170623348)

[3. ASR Micro Service 10](#_Toc170623349)

[Request 10](#_Toc170623350)

[Response 11](#_Toc170623351)

[4. RAG Retrieval Micro Service 11](#_Toc170623352)

[Request 11](#_Toc170623353)

[Response 11](#_Toc170623354)

[5. RAG Reranking Micro Service 12](#_Toc170623355)

[Request 12](#_Toc170623356)

[Response 12](#_Toc170623357)

This specification is used to define the Restful API of OPEA Mega Service for users to access, as long as the input and output definition of all OPEA Micro Services for developer to build OPEA Mega service.

# OPEA Mega Service API

OPEA Mega Service is the main entry user can access for a prebuilt GenAI application. Such GenAI application consists of single or several OPEA Micro Services chained as a DAG (Directed Acyclic Graph) and built as an execution workflow for developer to create complex applications.

## 1. List Services

List all supported services by the OPEA Mega Service.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | /v1/list\_service |

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  <service\_name>: <service\_description>  }  service\_name (**string**) – The endpoints or urls OPEA mega service is serving. For example, “/v1/RAG”. Please note some keywords like “v1/audio/speech”, “/v1/audio/transcriptions”, “/v1/embeddings”, “/v1/chat/completions” are reserved for openAI compatible Mega Service.  service\_description (**string**) – The detail usage description user used to access the specified endpoints or urls OPEA mega service is serving, including the request & post format and details. |
| 405 | {"error": "Retrieve service name wrongly."} |

## 2. List Configurable Parameters

List all configurable parameters for users to control the behavior of the OPEA Mega Service.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | /v1/list\_parameters |

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  <micro\_service\_name>: {  <parameter\_name>: data\_type,  ...  }  }  micro\_service\_name (**string**) – The micro service name in OPEA mega service, in which some parameters are configurable.  parameter\_name (**string**) – The configurable parameter name in OPEA mega service.  data\_type (**string**) – The supported data type, supporting “string” and “integer” now.  For example, {“/v1/llm\_generate”: {“max\_tokens”: “integer”}} |
| 405 | {"error": "Retrieve configurable parameter wrongly."} |

## 3. Embedding

Optional. Only exist if a single OPEA microservice which exposes “/micro\_service/embedding” interface is built as OPEA Mega service.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | /v1/embedding |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Params** | **Values** | **Required** |
| POST  POST  POST  POST | input  model  encoding\_format  dimensions | string  string  string  integer | required  deprecated  required  optional |

**input** Input text to embed, encoded as a string or array of tokens. To embed multiple inputs in a single request, pass an array of strings or array of token arrays. The input must not exceed the max input tokens for the model (8192 tokens for text-embedding-ada-002), cannot be an empty string, and any array must be 2048 dimensions or less.

**model** The ID of the model to use.

**encoding\_format** The format to return the embeddings in. Can be either float or base64.

**dimensions** The number of dimensions the resulting output embeddings should have. Only supported in text-embedding-3 and later models.

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "object": "list",  "data": [{  "object": "embedding",  "embedding": [  0.0023064255,  ...  ],  "index": 0  }],  "model": "text-embedding-ada-002",  "usage": {  "prompt\_tokens": 8,  "total\_tokens": 8  },  }  embedding (**float**) – The vector representation for given inputs.  index (**integer**) – The index of the embedding in the list of embeddings.  parameter\_name (**string**) – The configurable parameter name in OPEA mega service.  data\_type (**string**) – The supported data type, supporting “string” and “integer” now.  For example, {“llm”: {“max\_tokens”: “integer”}} |
| 405 | {"error": "Retrieve configurable parameter wrongly."} |

## 4. Chat

Optional. If a OPEA Mega service is built with this request url, it complies with below format.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | /v1/chat/completions |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Params** | **Values** | **Required** |
| POST  POST  POST  POST  POST  POST  POST  POST  POST  POST  POST  POST  POST  POST  POST  POST  POST  POST  POST | message  model  freqency\_penalty  logit\_bias  logprobs  top\_logprobs  max\_tokens  n  presence\_penalty  response\_format  seed  stop  stream  stream\_options  temperature  top\_p  tools  tool\_choice  user | array  string  integer  map  bool  integer  integer  integer  float  object  integer  string  bool  object  float  float  array  string  string | required  deprecated  optional  optional  optional  optional  optional  optional  optional  optional  optional  optional  optional  optional  optional  optional  optional  optional  optional |

**messages** A list of messages comprising the conversation. Refer to [the detail format](https://platform.openai.com/docs/api-reference/chat/create#chat-create-messages).

**model** The ID of the model to use.

**frequency\_penalty** Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat the same line verbatim.

**logit\_bias** Modify the likelihood of specified tokens appearing in the completion. Accepts a JSON object that maps tokens (specified by their token ID in the tokenizer) to an associated bias value from -100 to 100. Mathematically, the bias is added to the logits generated by the model prior to sampling. The exact effect will vary per model, but values between -1 and 1 should decrease or increase likelihood of selection; values like -100 or 100 should result in a ban or exclusive selection of the relevant token.

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "id": "chatcmpl-123",  "object": "chat.completion",  "created": 1677652288,  "model": "gpt-3.5-turbo-0125",  "system\_fingerprint": "fp\_44709d6fcb",  "choices": [{  "index": 0,  "object": "embedding",  "message": {  "role": "assistant",  "content": "\n\nHello there, how may I assist you today?",  },  "logprobs": null,  "finish\_reason": "stop",  }],  "usage": {  "prompt\_tokens": 9,  "completion\_tokens": 12,  "total\_tokens": 21  },  }  id (**string**) – A unique identifier for the chat completion.  choices (**array**) – A list of chat completion choices. Can be more than one if n is greater than 1.  created (**integer**) – The Unix timestamp (in seconds) of when the chat completion was created.  model (**string**) – The model used for the chat completion.  system\_fingerprint (**string**) – This fingerprint represents the backend configuration that the model runs with.Can be used in conjunction with the seed request parameter to understand when backend changes have been made that might impact determinism.  object (**string**) – The object type, which is always chat.completion.  usage (**object**) – Usage statistics for the completion request. |

## 5. Other operations

Check the usage description returned in [[list service]](#_Response) to know what other operations are supported by this OPEA Mega Service.

# OPEA Micro Service API

OPEA Micro Service is the building block of constructing any GenAI applications. The API in OPEA micro service is used by developers to construct OPEA Mega Service like a DAG chain and is invisible for end user.

## 1. Embedding Micro Service

The micro service is used to generate a vector representation of a given input.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | /v1/embeddings |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Params** | **Values** | **Required** |
| POST | input | string | required |
| POST | model | string | required |
| POST | encoding\_format | string | Optional, default to ‘float’ |
| POST | dimensions | integer | optional |
| POST | user | string | optional |

**input** Input text to embed, encoded as a string or array of tokens. To embed multiple inputs in a single request, pass an array of strings or array of token arrays. The input must not exceed the max input tokens for the model (8192 tokens for text-embedding-ada-002), cannot be an empty string, and any array must be 2048 dimensions or less.

**model** ID of the model to use.

**encoding\_format** The format to return the embeddings in. Can be either float or base64.

**Dimensions** The number of dimensions the resulting output embeddings should have.

**User** A unique identifier representing your end-user, which can help OpenAI to monitor and detect abuse.

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {    "object": "list",    "data": [      {        "object": "embedding",        "embedding": [          0.0023064255,          -0.009327292,          .... (1536 floats total for ada-002)          -0.0028842222,        ],        "index": 0      }    ],    "model": "text-embedding-ada-002",    "usage": {      "prompt\_tokens": 8,      "total\_tokens": 8    }  }  embedding (**list of float**) – The vector representation for given inputs. |
| 405 | {"error": "The request of getting embedding vector fails."} |

## 2. LLM Generation Micro Service

The micro service is used to provide LLM generation service.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | /v1/chat/completions |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Params** | **Values** | **Required** |
| POST | messages | array | required |
| POST | model | string | required |
| POST | frequency\_penalty | float | optional |
| POST | logit\_bias | map | optional |
| POST | logprobs | bool | optional |
| POST | top\_logprobs | integer | optional |
| POST | max\_tokens | integer | optional |
| POST | n | integer | optional |
| POST | presence\_penalty | float | optional |
| POST | response\_format | object | optional |
| POST | seed | integer | optional |
| POST | service\_tier | string | optional |
| POST | stop | string/array | optional |
| POST | stream | bool | optional |
| POST | stream\_options | object | optional |
| POST | temperature | float | optional |
| POST | top\_p | float | optional |
| POST | tools | array | optional |
| POST | tool\_choice | string/object | optional |

**messages** A list of messages comprising the conversation so far. Example Python code.

**model** ID of the model to use. See the model endpoint compatibility table for details on which models work with the Chat API.

**frequency\_penalty** Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat the same line verbatim.

**logit\_bias** Modify the likelihood of specified tokens appearing in the completion.Accepts a JSON object that maps tokens (specified by their token ID in the tokenizer) to an associated bias value from -100 to 100. Mathematically, the bias is added to the logits generated by the model prior to sampling. The exact effect will vary per model, but values between -1 and 1 should decrease or increase likelihood of selection; values like -100 or 100 should result in a ban or exclusive selection of the relevant token.

**Logprobs** Whether to return log probabilities of the output tokens or not. If true, returns the log probabilities of each output token returned in the content of message.

**top\_logprobs** An integer between 0 and 20 specifying the number of most likely tokens to return at each token position, each with an associated log probability. logprobs must be set to true if this parameter is used.

**max\_tokens** The maximum number of tokens that can be generated in the chat completion.The total length of input tokens and generated tokens is limited by the model's context length. Example Python code for counting tokens.

**n** How many chat completion choices to generate for each input message. Note that you will be charged based on the number of generated tokens across all of the choices. Keep n as 1 to minimize costs.

**presence\_penalty** Number between -2.0 and 2.0. Positive values penalize new tokens based on whether they appear in the text so far, increasing the model's likelihood to talk about new topics.

**response\_format** An object specifying the format that the model must output. Compatible with GPT-4o, GPT-4o mini, GPT-4 Turbo and all GPT-3.5 Turbo models newer than gpt-3.5-turbo-1106.Setting to { "type": "json\_schema", "json\_schema": {...} } enables Structured Outputs which ensures the model will match your supplied JSON schema. Learn more in the Structured Outputs guide.Setting to { "type": "json\_object" } enables JSON mode, which ensures the message the model generates is valid JSON.

**seed** This feature is in Beta. If specified, our system will make a best effort to sample deterministically, such that repeated requests with the same seed and parameters should return the same result. Determinism is not guaranteed, and you should refer to the system\_fingerprint response parameter to monitor changes in the backend.

**service\_tier** Specifies the latency tier to use for processing the request. This parameter is relevant for customers subscribed to the scale tier service:If set to 'auto', the system will utilize scale tier credits until they are exhausted.If set to 'default', the request will be processed using the default service tier with a lower uptime SLA and no latency guarentee. When this parameter is set, the response body will include the service\_tier utilized.

**stop** Up to 4 sequences where the API will stop generating further tokens.

**stream** If set, partial message deltas will be sent, like in ChatGPT. Tokens will be sent as data-only server-sent events as they become available, with the stream terminated by a data: [DONE] message. Example Python code.

**stream\_options** Options for streaming response. Only set this when you set stream: true.

**temperature** What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.We generally recommend altering this or top\_p but not both.

**top\_p** An alternative to sampling with temperature, called nucleus sampling, where the model considers the results of the tokens with top\_p probability mass. So 0.1 means only the tokens comprising the top 10% probability mass are considered. We generally recommend altering this or temperature but not both.

**tools** A list of tools the model may call. Currently, only functions are supported as a tool. Use this to provide a list of functions the model may generate JSON inputs for. A max of 128 functions are supported.

**tool\_choice** Controls which (if any) tool is called by the model. none means the model will not call any tool and instead generates a message. auto means the model can pick between generating a message or calling one or more tools. required means the model must call one or more tools. Specifying a particular tool via {"type": "function", "function": {"name": "my\_function"}} forces the model to call that tool.none is the default when no tools are present. auto is the default if tools are present.

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {    "id": "chatcmpl-123",    "object": "chat.completion",    "created": 1677652288,    "model": "gpt-4o-mini",    "system\_fingerprint": "fp\_44709d6fcb",    "choices": [{      "index": 0,      "message": {        "role": "assistant",        "content": "\n\nHello there, how may I assist you today?",      },      "logprobs": null,      "finish\_reason": "stop"    }],    "usage": {      "prompt\_tokens": 9,      "completion\_tokens": 12,      "total\_tokens": 21    }  } |
| 405 | {"error": "The request of LLM generation fails."} |

## 3. ASR Micro Service

The micro service is used to provide audio to text service.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | /v1/asr |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Params** | **Values** | **Required** |
| POST | url | docarray.AudioUrl | optional |
| POST | model\_name\_or\_path | string | optional |
| POST | Language | string | optional |

**url** The link to the audio.

**model\_name\_or\_path** The model used to do audio to text translation.

**language** The language that model prefer to detect. Default is "auto".

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "text": string  } |
| 405 | {"error": "The request of ASR fails."} |

## 4. RAG Retrieval Micro Service

The micro service is used to provide RAG retrieval service. It’s usually after embedding micro sevice and before RAG reranking micro service to build a RAG Mega service.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | /v1/rag\_retrieval |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Params** | **Values** | **Required** |
| POST | text | string | required |
| POST | embedding | list of float | required |

**text** The input string to query.

**embedding** The list of float for text as vector representation.

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "retrieved\_docs": list of string,  "initial\_query": string,  "json\_encoders": [{  "text": "I am the agent of chatbot. What can I do for you?",  },  ...  ]  } |
| 405 | {"error": "The request of ASR fails."} |

## 5. RAG Reranking Micro Service

The micro service is used to provide RAG reranking service. It’s usually after RAG retrieval and before LLM generation micro service.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | /v1/rag\_reranking |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Params** | **Values** | **Required** |
| POST | retrieved\_docs | list of string | required |
| POST | initial\_query | string | required |
| POST | json\_encoders | list of float | required |

**retrieved\_docs** The docs to be retrieved.

**initial\_query** The string to query.

**json\_encoders** The json encoder used.

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "query": string,  "doc": [{  "text": "I am the agent of chatbot. What can I do for you?",  },  ...  ]  } |
| 405 | {"error": "The request of ASR fails."} |