Kandinsky 2.2

Kandinsky 2.2 is an open source Gen AI multilingual text-to-image Latent Diffusion Model(LDM). In this API, when a text prompt– usually a phrase– is provided to the AI model, an image is generated according to the prompt. For example, if the input text prompt is “A moss covered astronaut with a black background” then the output is an image with an astronaut covered in moss.

Note: This API reference document is a sample for one out of four modules . For more information on remaining modules refer to ai-forever/kandinsky-2.2 github repository.

Authentication

Kandinsky uses a `REPLICATE` access token to authorize your application. Create a `REPLICATE` account to get an API token. For more information refer to `REPLICATE` website

curl setup

Set the REPLICATE environment

```
export REPLICATE_API_TOKEN=<paste-your-token-here>
```

```
curl --silent --show-error https://api.replicate.com/v1/predictions \
      --request POST \
      --header "Authorization: Bearer $REPLICATE_API_TOKEN" \
      --header "Content-Type: application/json" \
      --header "Prefer: wait" \
```

```
      --data @- <<-EOM
{
      "version":
"ad9d7879fbffa2874e1d909d1d37d9bc682889cc65b31f7bb00d2362619f194a",
      "input": {
      "width": 1024,
      "height": 1024
      }
}
EOM
```

Python setup

Install REPLICATE's python library

```
pip install replicate
```

```
import replicate

input = {
    "width": 1024,
    "height": 1024
}

output = replicate.run(

"ai-forever/kandinsky-2.2:ad9d7879fbffa2874e1d909d1d37d9bc682889cc65b31f7bb
00d2362619f194a",
    input=input
)
for index, item in enumerate(output):
    with open(f"output_{index}.png", "wb") as file:
        file.write(item.read())
#=> output_0.png written to disk
```

To set up a python environment refer to the instructions.
For more information, refer to authentication to use Kandinsky.

API Reference

Kandinsky primarily has four models:

- `prior`
- `text2image`
- `image fuse`
- `inpainting`

`text2image`

The `text2image` API creates images from the textual prompts provided by the user. It is a decoding diffusion model that converts text embeddings(which are created using an encoder) into image embeddings. These embeddings are later deployed into other Kandinksy's image models such as "`Image-to-Image`", "`Inpainting`" and "`ControlNet-depth`" to modify and increase the accuracy of the image predictions and output.

Run the model

Base URL

```
https://api.replicate.com/v1/predictions
```

curl

```
export REPLICATE_API_TOKEN=<paste-your-token-here>
curl -s -X POST \
  -H "Authorization: Bearer $REPLICATE_API_TOKEN" \
  -H "Content-Type: application/json" \
  -H "Prefer: wait" \
  -d $'{
    "version":
"ad9d7879fbffa2874e1d909d1d37d9bc682889cc65b31f7bb00d2362619f194a",
    "input": {
      "width": 1024,
      "height": 1024,
      "prompt": "A moss covered astronaut with a black background",
      "num_outputs": 1,
```

```
      "output_format": "webp",
      "num_inference_steps": 75,
      "num_inference_steps_prior": 25
    }
  }' \
  https://api.replicate.com/v1/predictions
```

Python

```
pip install diffusers transformers accelerate

from diffusers import AutoPipelineForText2Image
import torch

pipe =
AutoPipelineForText2Image.from_pretrained("kandinsky-community/kandinsky-2-
2-decoder", torch_dtype=torch.float16)
pipe = pipe.to("cuda")

prompt = "A moss covered astronaut with a black background"
negative_prompt = "low quality, bad quality"

image = pipe(prompt=prompt, negative_prompt=negative_prompt,
prior_guidance_scale =1.0, height=768, width=768).images[0]
image.save("portrait.png")
```

Note: Refer to Kandinsky 2.2 repository for the list of modules and Kandinsky's community for all the other module implementations.

Parameter list

Input	Type	Parameter	Description
	enum	width	Required width of the output image. Example: 1024

	enum	`Height`	`Required height of the output image. Example: 1024`
	string	`Prompt`	`An input string. Example: “A moss covered astronaut with a black background”`
	int	`Num_outputs`	`Number of images in the output`
	enum	`Output_format`	The output image format. Example: “jpeg”,”png”
	int	`Num_inference_steps`	The number of denoising or reducing the distortion steps.
	int	`num_inference_steps_prior`	The number of denoising steps for prior model.

REST Resources

predictions.create	Create a prediction and get the output.
predictions.get	Get a prediction
predictions.cancel	Cancel a prediction
predictions.list	Lists the predictions

Request

`predictions.create`

```
POST/predictions
```

curl

```
curl --silent --show-error https://api.replicate.com/v1/predictions \
      --request POST \
      --header "Authorization: Bearer $REPLICATE_API_TOKEN" \
      --header "Content-Type: application/json" \
      --header "Prefer: wait" \
      --data @- <<-EOM
{
      "version":
"ad9d7879fbffa2874e1d909d1d37d9bc682889cc65b31f7bb00d2362619f194a",
      "input": {
      "width": 1024,
      "height": 1024
      }
}
EOM
```

Python

```
import replicate

input = {
    "width": 1024,
    "height": 1024
}

output = replicate.run(

"ai-forever/kandinsky-2.2:ad9d7879fbffa2874e1d909d1d37d9bc682889cc65b31f7bb
00d2362619f194a",
    input=input
)
for index, item in enumerate(output):
    with open(f"output_{index}.png", "wb") as file:
        file.write(item.read())
#=> output_0.png written to disk
```

Node.js

```
import { writeFile } from "fs/promises";
import Replicate from "replicate";
const replicate = new Replicate();

const input = {
    width: 1024,
    height: 1024
};

const output = await
replicate.run("ai-forever/kandinsky-2.2:ad9d7879fbffa2874e1d909d1d37d9bc682
889cc65b31f7bb00d2362619f194a", { input });
for (const [index, item] of Object.entries(output)) {
  await writeFile(`output_${index}.png`, item);
}
//=> output_0.png written to disk
```

predictions.get

```
GET/predictions/{prediction_id}
```

Parameters

Name	Description
prediction_id	The ID of the prediction to retrieve.

curl

```
curl --silent https://api.replicate.com/v1/predictions/{prediction_id} \
    --request GET \
    --header "Authorization: Bearer $REPLICATE_API_TOKEN"
```

Python

```
import replicate

prediction = replicate.predictions.get(prediction_id)
print(prediction)
#=> Prediction(id="xyz...", status="successful", ... )
```

predictions.cancel

```
POST/predictions/{prediction_id}/cancel
```

curl

```
curl --silent
https://api.replicate.com/v1/predictions/{prediction_id}/cancel \
    --request POST \
    --header "Authorization: Bearer $REPLICATE_API_TOKEN"
```

Python

```
import replicate

prediction = replicate.predictions.get(prediction_id)
prediction.cancel()
print(prediction)
#=> Prediction(id="xyz...", status="canceled", ... )
```

Sample Request

Python

```
output = replicate.run(
```

```
"ai-forever/kandinsky-2.2:ad9d7879fbffa2874e1d909d1d37d9bc682889cc65b31f7bb
00d2362619f194a",
    input={
        "width": 1024,
        "height": 1024,
        "prompt": "A moss covered astronaut with a black background",
        "num_outputs": 1,
        "output_format": "webp",
        "num_inference_steps": 75,
        "num_inference_steps_prior": 25
    }
)
print(output)
```

Sample Response

JSON


```
{
  "completed_at": "2024-09-12T22:23:00.041096Z",
  "created_at": "2024-09-12T22:22:51.044455Z",
  "data_removed": false,
  "error": null,
  "id": "3tsx2ijbtsa47m4mwbifyiglxm",
  "input": {
    "width": 1024,
    "height": 1024,
    "prompt": "A moss covered astronaut with a black background",
    "num_outputs": 1,
    "num_inference_steps": 75
  },
  "logs": "Using seed: 4697\n  0%|          | 0/25 [00:00<?, ?it/s]\n
20%|██        | 5/25 [00:00<00:00, 40.13it/s]\n 40%|████      | 10/25
[00:00<00:00, 40.19it/s]\n 60%|██████    | 15/25 [00:00<00:00,
40.05it/s]\n 80%|████████  |",
  "metrics": {
    "predict_time": 9.029294,
    "total_time": 8.996641
  },
```

```
  "output": [

"https://replicate.delivery/pbxt/Lca3IEjcKoJBBVS6ajROkK37sDzPsmjYxIcFzxPZp65wZzTE/out-0.png"
  ],
  "started_at": "2023-07-12T22:22:51.011802Z",
  "status": "succeeded",
  "urls": {
    "get":
"https://api.replicate.com/v1/predictions/3tsx2ijbtsa47m4mwbifyiglxm",
    "cancel":
"https://api.replicate.com/v1/predictions/3tsx2ijbtsa47m4mwbifyiglxm/cancel
"
  },
  "version":
"424befb1eae6af8363edb846ae98a11111a39740988baebd279d73fe3ecc92c2"
}
```


References:

1. Replicate:: https://replicate.com/ai-forever/kandinsky-2.2
2. Github: https://github.com/ai-forever/Kandinsky-2/blob/main/README.md
3. Huggingface: https://huggingface.co/kandinsky-community/kandinsky-2-2-prior#text-to-image
4. Licence: Apache Licence 2.0