STABLE DIFFUSION

I M A G E   T O   P R O M P T S

남승우 신소연 안세정 정건우

CONTENTS

01 Task 설명

02 BLIP2 모델 설명

03 BLIP2 모델 구현

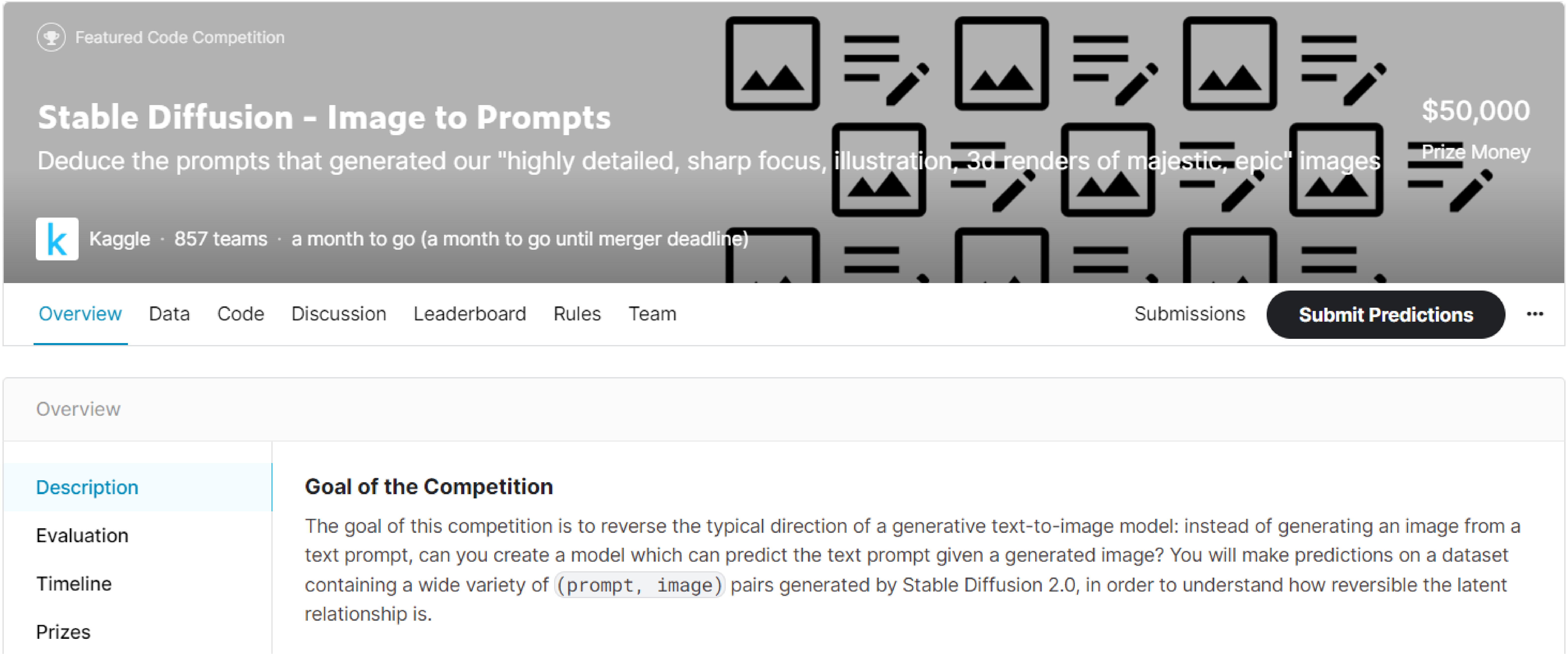
04 성능 평가

05 한계점 및 활용

01

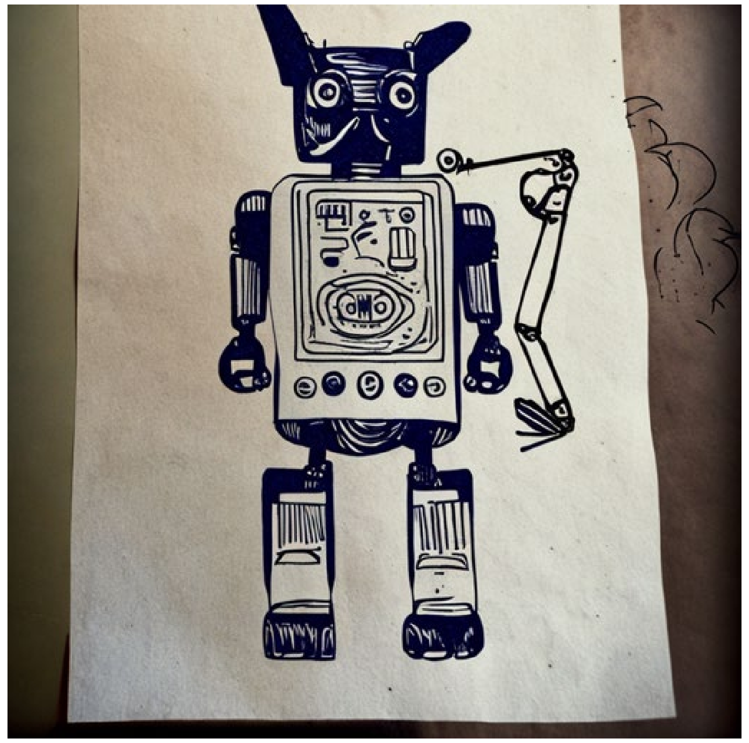
TASK 설명

1. TASK 설명


Featured Code Competition
Stable Diffusion - Image to Prompts
Deduce the prompts that generated our "highly detailed, sharp focus, illustration, 3d renders of majestic, epic" images
$50,000
Prize Money
Kaggle · 857 teams · a month to go (a month to go until merger deadline)
Overview
Data
Code
Discussion
Leaderboard
Rules
Team
Submissions
Submit Predictions
Overview
Description
Evaluation
Timeline
Prizes
Goal of the Competition
The goal of this competition is to reverse the typical direction of a generative text-to-image model: instead of generating an image from a text prompt, can you create a model which can predict the text prompt given a generated image? You will make predictions on a dataset containing a wide variety of (prompt, image) pairs generated by Stable Diffusion 2.0, in order to understand how reversible the latent relationship is.

1. TASK 설명

Image Captioning



a thundering retro robot crane inks on parchment with a droopy french bulldog

an astronaut standing on a engaging white rose, in the midst of by ivory cherry blossoms

02

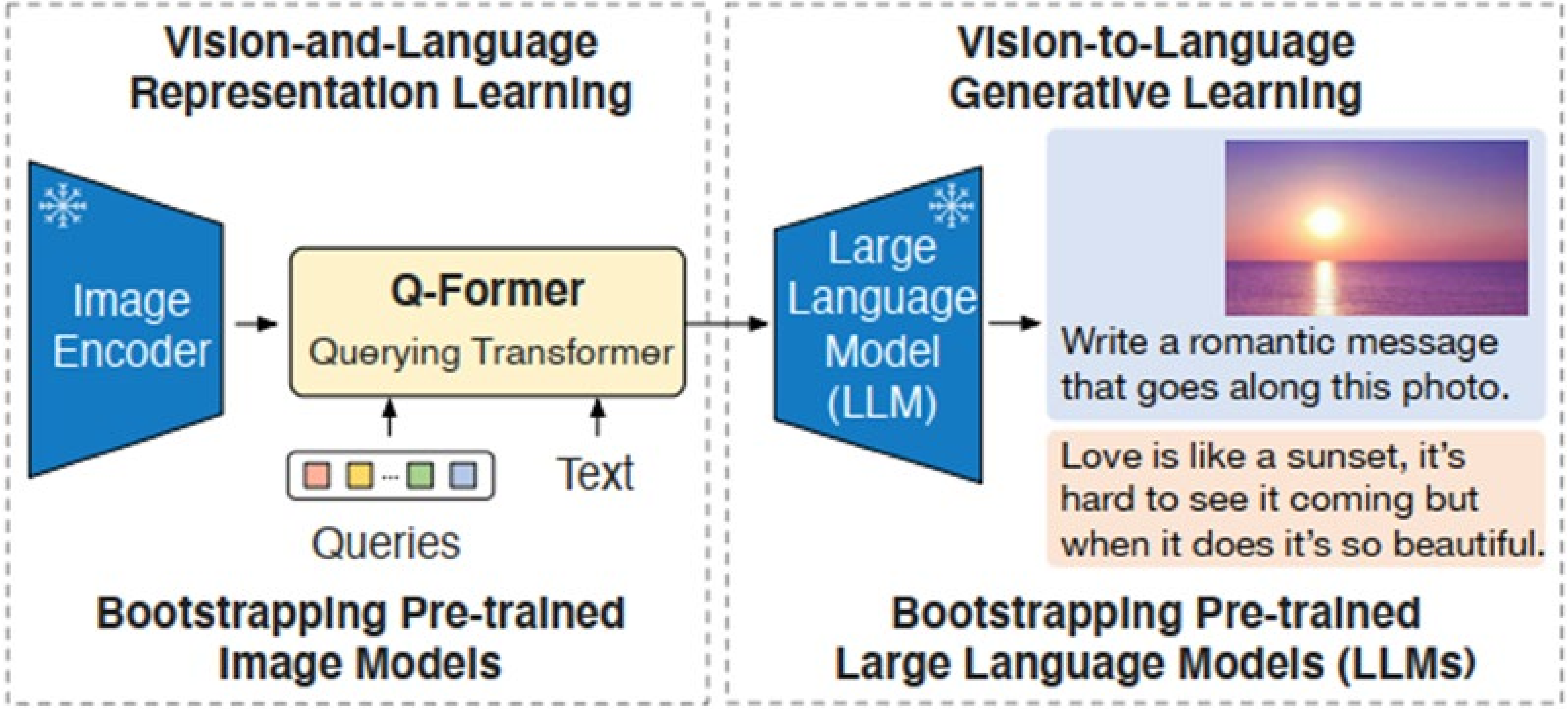
BLIP2 모델 설명

2. BLIP2 모델 설명

Image-to-text

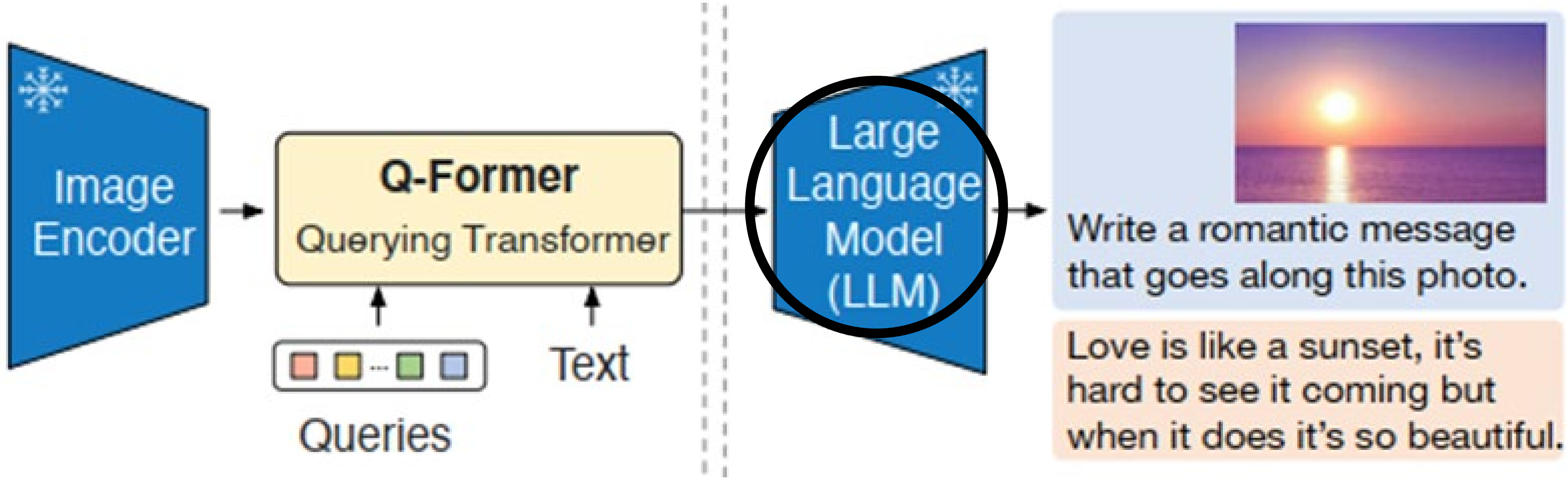
1) Frozen Pre-trained Image Encoder (Image representation)

2) Frozen Large Language Model (Text generation)



2. BLIP2 모델 설명

modality gap 해결 <Visual features & text features align>

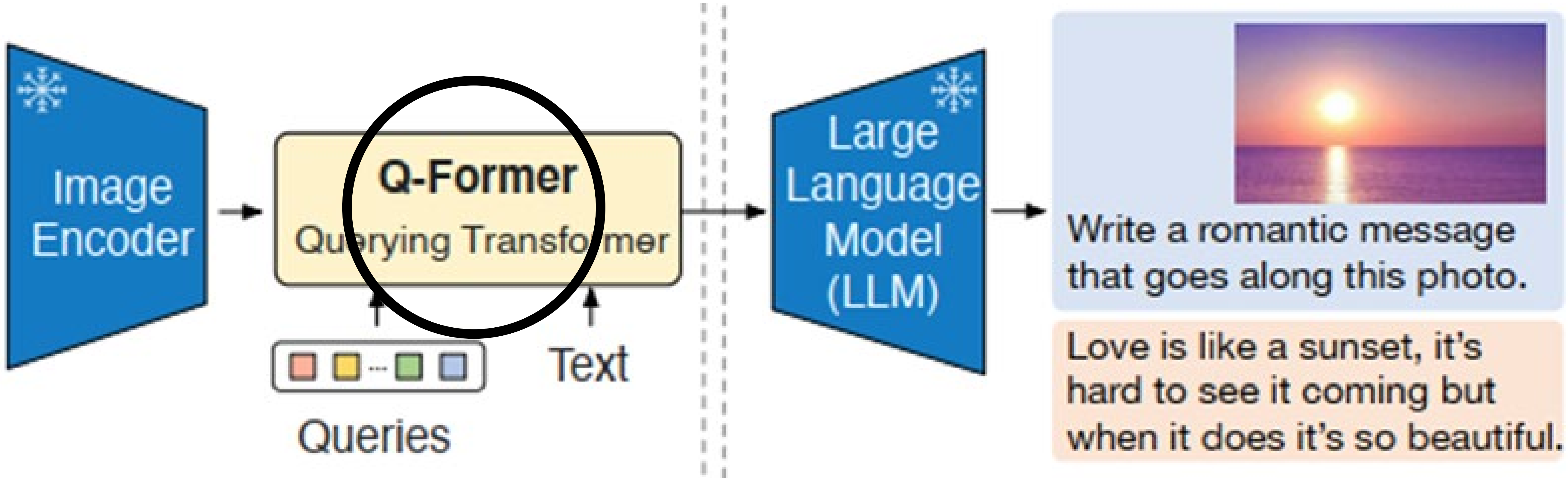


문제는

- LLM은 Unimodal language model : 사전 학습 과정에서 image 정보를 받지 않음
- Frozen LLM : 더 이상 학습하지 않음

2. BLIP2 모델 설명

modality gap 해결 <Visual features & text features align>

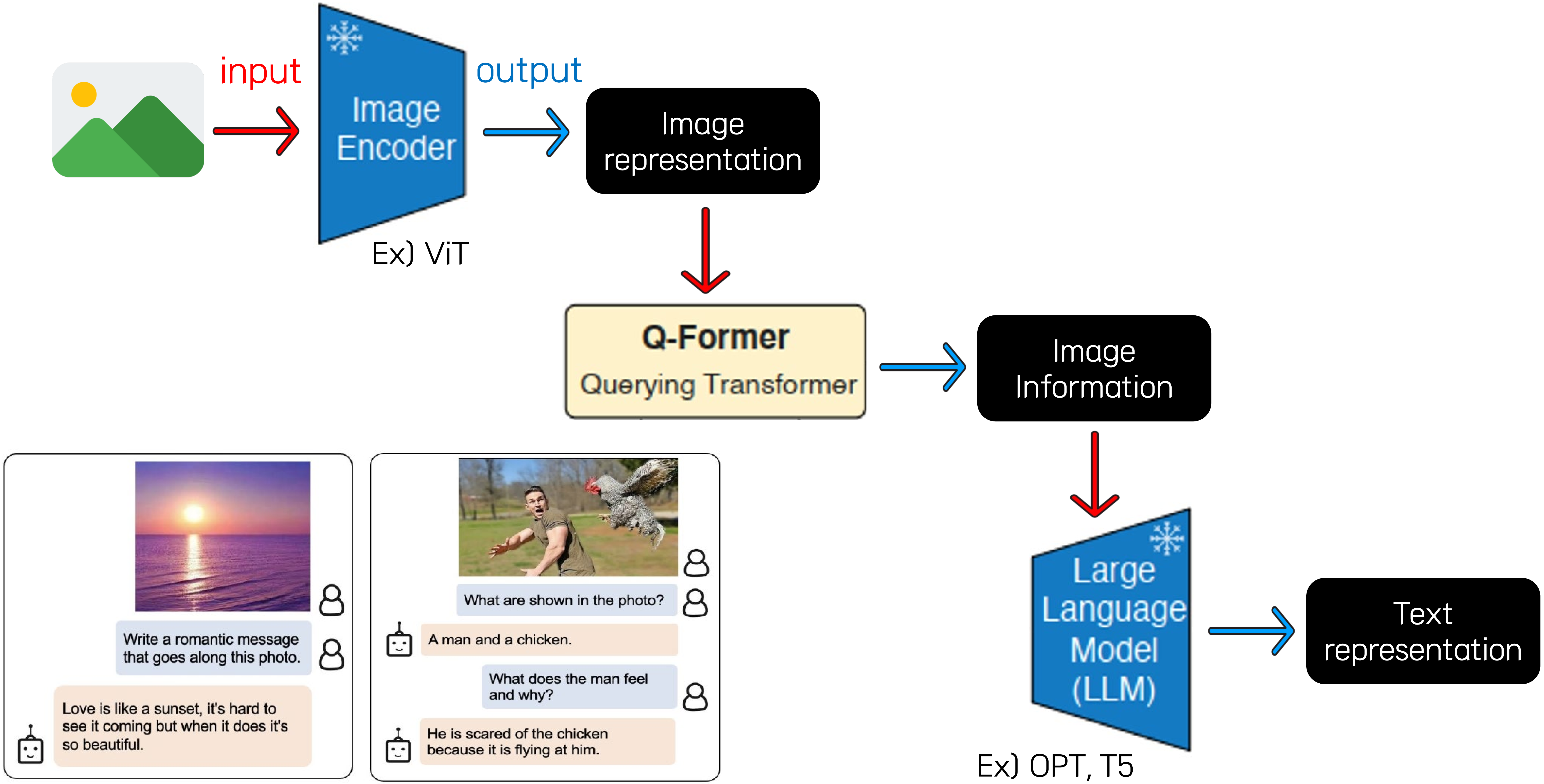


문제는

- LLM은 Unimodal language model : 사전 학습 과정에서 image 정보를 받지 않음
- Frozen LLM : 더 이상 학습하지 않음

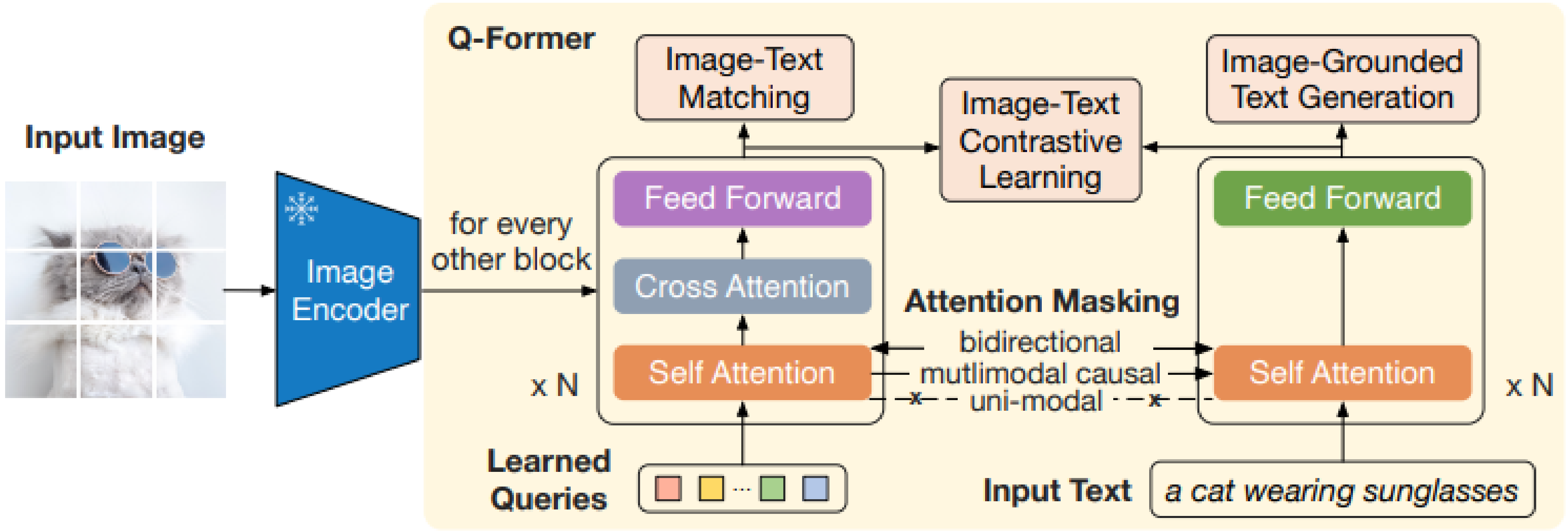
➡ Q-Former (Querying Transformer) 제시

2. BLIP2 모델 설명 : 전체 구조



2. BLIP2 모델 설명 : Q-Former

Q-former는 Image encoder (ex.ViT)와 LLM (ex.OPT, T5)의 **modality gap을 줄이는** 징검다리 역할



stage 1) Vision-language Representation learning

: frozen image encoder에서 text와 관련이 있는 visual features를 extraction

stage 2) Vision-language Generative learning

: stage 1을 기반으로, 주어진 이미지에 적합한 text를 생성

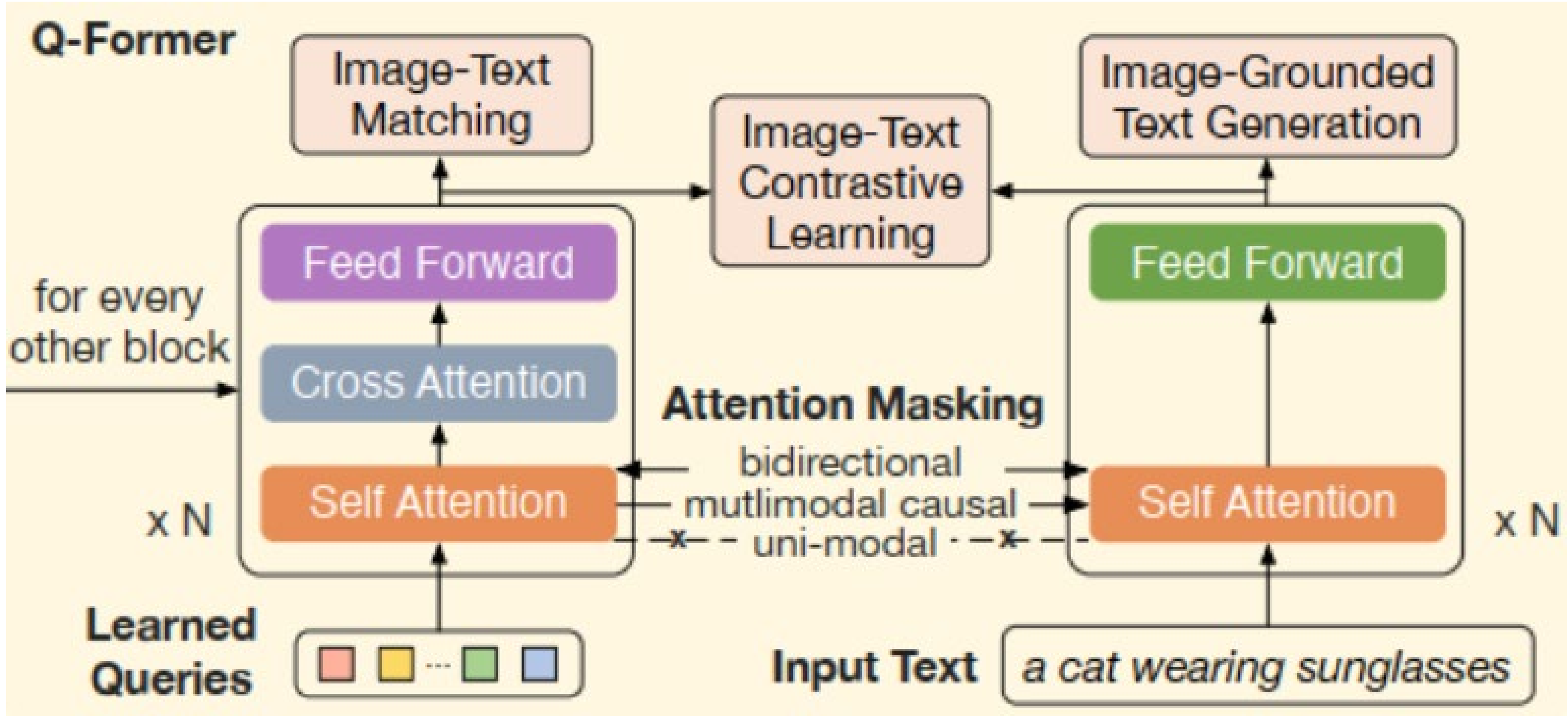
2. BLIP2 모델 설명 : Q-Former

Stage 1 : Representation Learning

Frozen image encoder에서 text와 관련이 있는 visual features를 extraction

3가지 objective를 **jointly optimize**하는 과정

- Image-Text Contrastive Learning (ITC)
- Image-grounded Text Generation (ITG)
- Image-Text Matching (ITM)



2. BLIP2 모델 설명 : Q-Former

- Image-Text Contrastive Learning (ITC)

: Image representation과 text representation의 **유사도**가 가장 높은 pair를 선정

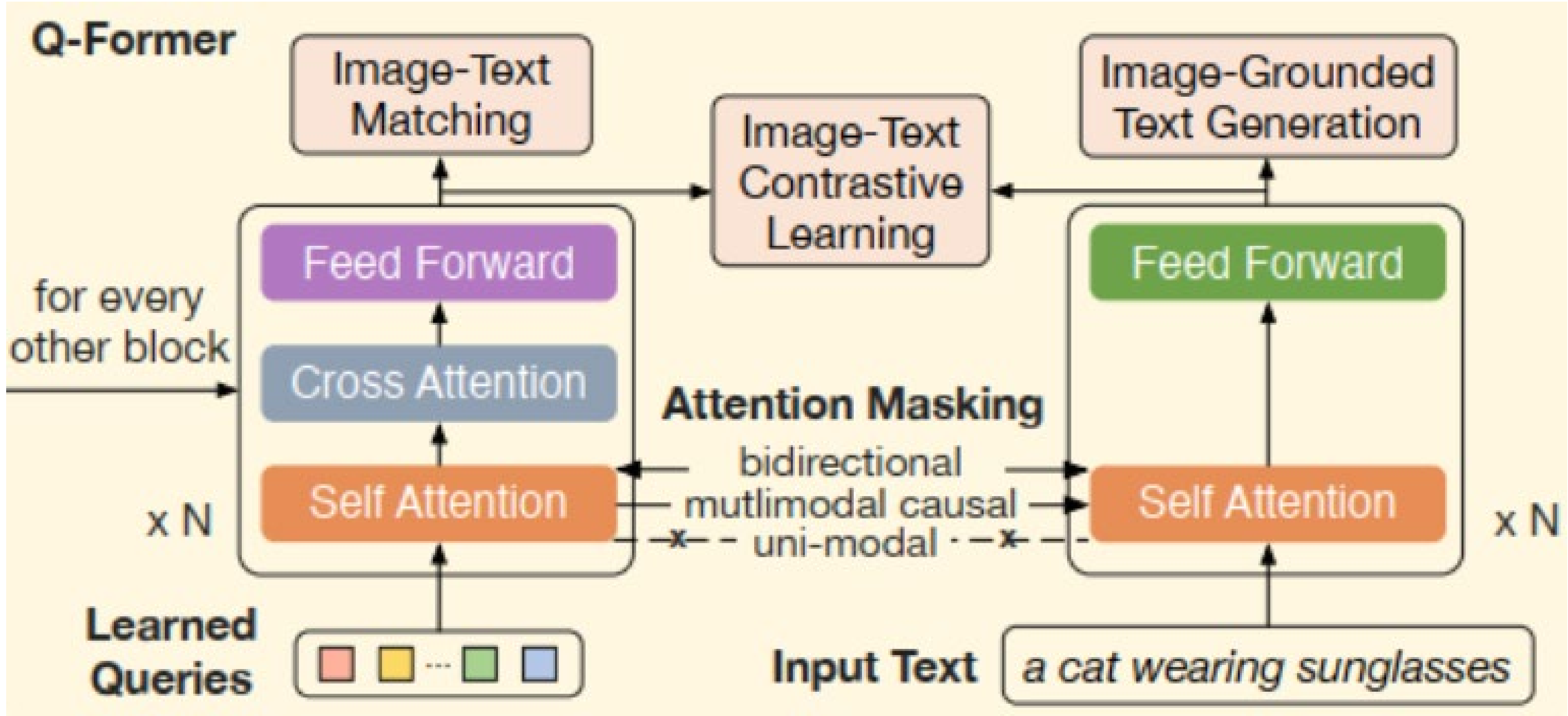
- Image-grounded Text Generation (ITG)

: Image representation을 잘 설명하는 **text 생성**

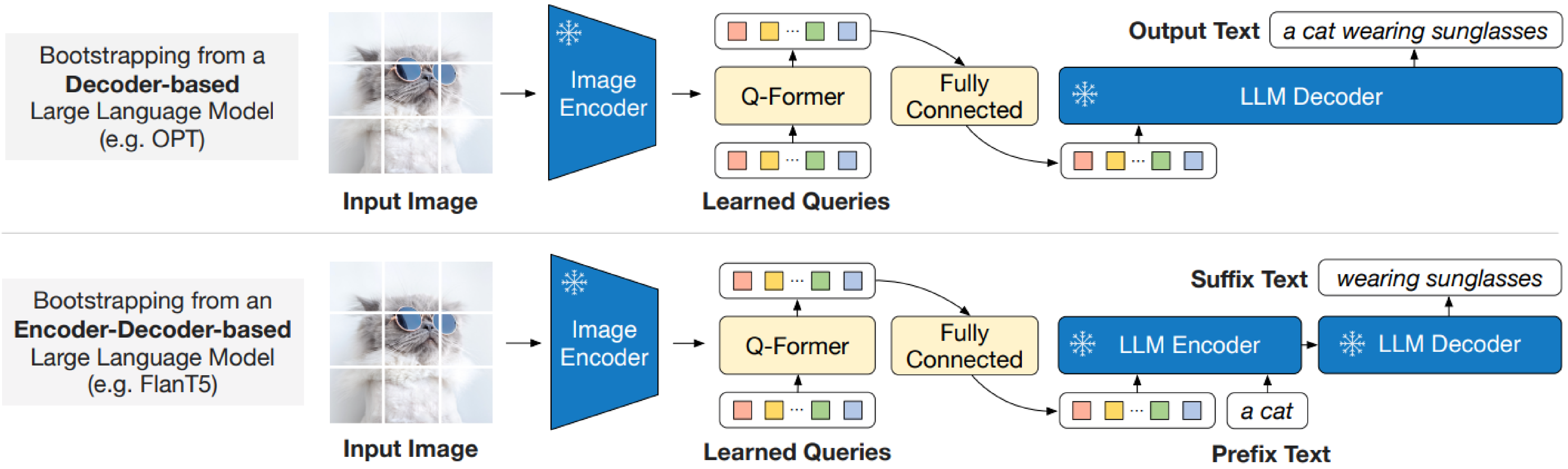
- Image-Text Matching (ITM)

: Image와 text representation이

positive(match)한지 예측할 수 있도록 학습



2. BLIP2 모델 설명 : Q-Former



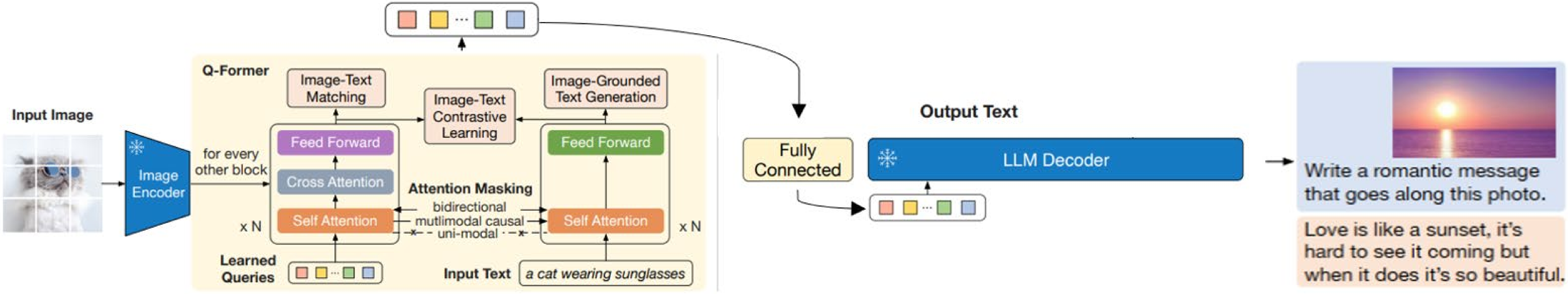
Stage 2 : Generative Learning

- Q-Former의 output query는 Fully Connected Layer를 통해 LLM로 전달됨
- Dimension을 LLM의 text embedding의 dimension과 같게 만들기 위해 FC Layer 사용
- Q-Former가 visual representation에서 **관련도가 높은** 정보를 추출하도록 학습되었으므로 Image 정보를 학습한 적이 없는 LLM도 Q-former 덕분에 좋은 text를 만들어낼 수 있음

2. BLIP2 모델 설명 : Q-Former

결론적으로 Q-Former의 역할은:

Image Encoder에서 추출한 visual features를
LLM이 해석할 수 있도록 text features에 align



2. BLIP2 모델 설명 : Q-Former

Image Encoder (ex. ViT) 와 LLM (ex. OPT, T5) 를 연결하는 이유

Image Encoder와 LLM을 연결만 할 수 있으면
둘 다 frozen 상태로 가져오면 되고,
parameters를 학습시킬 필요가 없다

〈 연결 : Q-Former 〉

기존 SOTA 모델보다
적은 개수의 parameters를 학습시켜도
더 좋은 성능을 낼 수 있다!!

BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models

Models	#Trainable Params	Open-sourced?	**Visual Question Answering** VQAv2 (test-dev)	**Image Captioning** NoCaps (val)		**Image-Text Retrieval** Flickr (test)	
			VQA acc.	CIDEr	SPICE	TR@1	IR@1
BLIP (Li et al., 2022)	583M	✓	-	113.2	14.8	96.7	86.7
SimVLM (Wang et al., 2021b)	1.4B	✗	-	112.2	-	-	-
BEIT-3 (Wang et al., 2022b)	1.9B	✗	-	-	-	94.9	81.5
Flamingo (Alayrac et al., 2022)	10.2B	✗	56.3	-	-	-	-
BLIP-2	188M	✓	**65.0**	**121.6**	**15.8**	**97.6**	**89.7**

03

BLIP2 모델 구현

3. BLIP2 모델 구현

Image

CLIP

Top related keywords:
(Tensor Form)
[[0.123456789·········],
[-0.63543645·········..],
·································..]]

CLIP interrogator

Top related keywords:
an illustration of,
sumatraism,
mmmmm,
buttercup eating pizza,
pastry lizard

BLIP2

Caption:
a cartoon dinosaur with a piece of cheese on itan illustration of

Prompt

Final prompt:
a cartoon dinosaur with a piece of cheese on itan illustration of
an illustration of,
sumatraism,
mmmmm,
buttercup eating pizza,
pastry lizard

3. BLIP2 모델 구현

Architecture (Colab-based)

필요한 패키지 설치 & 임포트

```
#install the package
!pip install open_clip_torch
!pip install clip-interrogator==0.6.0
!pip install -U sentence-transformers
```

```
# import packages
import torch
from PIL import Image
import open_clip
import inspect
import importlib
from clip_interrogator import clip_interrogator
from clip_interrogator import Config, Interrogator
from pathlib import Path
from sentence_transformers import SentenceTransformer, models
```

```
#install the dataset of competition
from google.colab import files
files.upload()
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
!kaggle competitions download -c stable-diffusion-image-to-prompts
```

```
!unzip -o '/content/stable-diffusion-image-to-prompts.zip' -d '/content/'
```

3. BLIP2 모델 구현

Competition Dataset에서 Sample image, Sample submission 다운로드

```python
import pandas as pd
import numpy as np
import os

#bring images of sample submission file
sample_submission = pd.read_csv('/content/sample_submission.csv', index_col = 'imgId_eId')
images = os.listdir('/content/images')
image_ids = [i.split('.')[0] for i in images]
EMBEDDING_LENGTH = 384
eIds = list(range(EMBEDDING_LENGTH))
imgId_eId = [
    '_'.join(map(str, i)) for i in zip(
        np.repeat(image_ids, EMBEDDING_LENGTH), # [인덱스 0부터 6 384번 반복]
        np.tile(range(EMBEDDING_LENGTH), len(image_ids)) # [0 ~ 383, 0 ~ 383, ......]
    )
]
def make_batches(l, batch_size=16):
    for i in range(0, len(l), batch_size):
        yield l[i:i + batch_size]
```

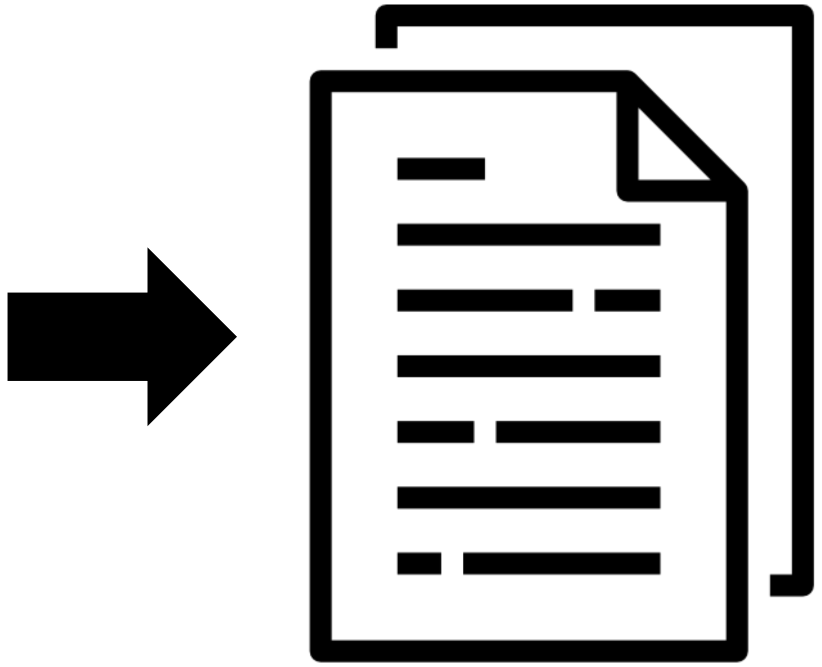
3. BLIP2 모델 구현

CLIP pre-trained model 선택해 preprocessor, model, token 생성

```
#selecting the CLIP model - ViT-g-14/laion2b_s34b_b88k
model, _, preprocess = open_clip.create_model_and_transforms('ViT-g-14',
                                                             pretrained = 'laion2b_s34b_b88k')
tokenizer = open_clip.get_tokenizer('ViT-g-14')
st_model = SentenceTransformer('sentence-transformers/all-MiniLM-L6-v2')
```

CLIP encoding으로 인해 생성되는
embedding tensor와 매치될 wordset 생성

```
ci = Interrogator(Config(clip_model_name = 'ViT-g-14/laion2b_s34b_b88k'))
mediums_features_array = torch.stack([torch.from_numpy(t) for t in ci.mediums.embeds])
movements_features_array = torch.stack([torch.from_numpy(t) for t in ci.movements.embeds])
flavors_features_array = torch.stack([torch.from_numpy(t) for t in ci.flavors.embeds])
```



3. BLIP2 모델 구현

미리 학습된 CLIP model로 Sample images encoding

```python
BATCH_SIZE = 32
clip_text = []
cos = torch.nn.CosineSimilarity(dim=1)
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
for batch in make_batches(images, BATCH_SIZE):
  images_batch = []
  for i, image in enumerate(batch):
    images_batch.append(preprocess(Image.open('/content/images/'+image).convert('RGB')).unsqueeze(0))
  images_batch = torch.cat(images_batch, 0)

  with torch.no_grad(), torch.cuda.amp.autocast():
    image_features = model.encode_image(images_batch)
    image_features /= image_features.norm(dim = -1, keepdim = True)

  for i in range(len(image_features)):
    medium = [ci.mediums.labels[i] for i in cos(image_features[i], mediums_features_array).topk(1).indices][0]
    movement = [ci.movements.labels[i] for i in cos(image_features[i], movements_features_array).topk(1).indices][0]
    flaves = ', '.join([ci.flavors.labels[i] for i in cos(image_features[i], flavors_features_array).topk(3).indices])
    prompt = f'{medium}, {movement}, {flaves}'
    clip_text.append(prompt)
for i in clip_text:
  print(i)
```

```
a woodcut, art nouveau, whorl, carved wood, swirl
a digital painting, context art, planet arrakis, crater, looking down at a massive crater
digital art, digital art, the mighty donut, at the counter, donut
a storybook illustration, digital art, nachosaurus, "a dinosaur market, pastry lizard
digital art, conceptual art, american astronaut in the forest, astronaut walking, lonely astronaut
a screenprint, lowbrow, robot!, rabbit robot, robot
a detailed painting, magic realism, oil canvas of lucifer, epic surrealism 8k oil painting, thomas blackshear and moebius
```

encoding된 image tensor와
코사인 유사도가 가장 높은
wordset index 5개 추출

wordset에서 해당 index 위치의
word 가져와 prompt 생성

woodcut
art nouveau
whorl
carved wood
swirl

3. BLIP2 모델 구현

BLIP-2 pretrained model 선택해 preprocessor, model 생성

```python
from transformers import Blip2Processor, Blip2ForConditionalGeneration

processor = Blip2Processor.from_pretrained('salesforce/blip2-flan-t5-xl')
model = Blip2ForConditionalGeneration.from_pretrained('salesforce/blip2-flan-t5-xl')
```

```python
model.to(device)
BATCH_SIZE = 16
cap_list = []
for ix, batch in enumerate(make_batches(images, BATCH_SIZE)):
  images_batch = []
  for i, image in enumerate(batch):
    images_batch.append(Image.open('/content/images/'+image).convert('RGB'))
  pixel_values = processor(images = images_batch, return_tensors = 'pt').pixel_values.to(device)
  out = model.generate(pixel_values = pixel_values, max_length = 20, num_return_sequences = 5,
                       num_beams = 5, min_length = 5)
  prompts = processor.batch_decode(out, skip_special_tokens = True)
```

미리 학습된 BLIP-2 기반
image를 encode – text tensor로
decode

A circular piece of wood with a spiral design on it
A circular piece of wood with a spiral design
A circular piece of wood with a spiral pattern on it
A circular piece of wood with a spiral on it
A circular piece of wood with a spiral pattern

3. BLIP2 모델 구현

```
for i in range(len(images_batch)):
    for j in range(5):
        caption = prompts[i * 5 + j]
        prompt = caption + clip_text[BATCH_SIZE * ix + i]
        cap_list.append(prompt)
for i in cap_list:
  print(i)
```

합쳐진 caption + prompt 리스트에 저장

높은 유사도 가진 image별 5개의 caption에 기존 clip prompt concatenate

woodcut
art nouveau
whorl
carved wood
swirl

① A circular piece of wood with a spiral design on it
② A circular piece of wood with a spiral design
③ A circular piece of wood with a spiral pattern on it
④ A circular piece of wood with a spiral on it
⑤ A circular piece of wood with a spiral pattern

① +
② +
③ +
④ +
⑤ +

Ex)

A circular piece of wood with a spiral design on it woodcut art nouveau whorl carved wood swirl

3. BLIP2 모델 구현

text가 들어있는 list sentence transformer로 **tensor 변환**

```
# Convert text to embeddings
submission_custom = st_model.encode(cap_list).flatten()
submission_custom = np.reshape(submission_custom, (-1, 5, 384)).mean(1).flatten()
print(len(submission_custom))
submission = (np.array(submission_custom))
print(len(submission))
print(len(imgId_eId))
submission = pd.DataFrame({'imgId_eId': imgId_eId,
                           'val' : submission})
```

Image별 5개의 높은 유사도를 가진 text ··· 〉 concatenate된 5개

image별 한 개의 tensor만 반환해야 하므로
5개의 **text tensor Average 변환** ··· 〉 5개의 평균

image별 tensor 값 지정해주어
submission.csv 파일 제작 ··· 〉 제출!!

04

성능 평가

4. 성능 평가

```
images = os.listdir('/content/images')
imgIds = [i.split('.')[0] for i in images]
EMBEDDING_LENGTH = 384
eIds = list(range(EMBEDDING_LENGTH))

imgId_eId = [
    '_'.join(map(str, i)) for i in zip(
        np.repeat(imgIds, EMBEDDING_LENGTH),
        np.tile(range(EMBEDDING_LENGTH), len(imgIds)))]

assert sorted(imgId_eId) == sorted(submission.imgId_eId)
ground_truth = pd.read_csv('/content/prompts.csv')
ground_truth = pd.merge(pd.DataFrame(imgIds, columns = ['imgId']), ground_truth,
                        on = 'imgId', how = 'left')
ground_truth_embeddings = st_model.encode(ground_truth.prompt).flatten()
gte = pd.DataFrame(
    index = imgId_eId,
    data = ground_truth_embeddings,
    columns = ['val']
).rename_axis('imgId_eId')

from scipy import spatial
vec1 = gte['val']
vec2 = submission['val']
cos_sim = 1 - spatial.distance.cosine(vec1, vec2)
print(cos_sim)
```

```
0.5331262946128845
```

0.5331262946128845

05

한계점 및 활용

5. 한계점 및 활용

1. Tokenizer 호환 x
- Encoder: CLIP Interrogator & Decoder: KoBERT
-각각 다른 모델을 불러왔는데, 서로 다른 tokenizer를 사용하기 때문에 호환되지 않았다

2. 용량 문제 (CPU, GPU, RAM 등)
- kaggle에서는 BLIP2 모델을 사용하지 못하고 BLIP1 모델을 사용해 학습시켰다
- colab 환경에서도 BLIP2의 가장 용량이 큰 Pre-trained dataset은 학습이 어려웠다

3. 표현력의 한계
- Clip interrogator에 있는 mediums, flavors, movement 단어 set이 큰 편이 아니어서 표현에 부족함이 있었다

5. 한계점 및 활용

한국어 Image Captioning 모델 (KoCLIP, KoBLIP)

KoCLIP

- 공개된 모델이 있다: prompt가 주어지면 빈칸에 들어갈 단어만 예측하는 정도
  - ‘이것은 {{}}이다.’

KoCLIP

Open in Streamlit | Open in Colab

This repository contains code for KoCLIP, a Korean port of OpenAI's CLIP. This project was conducted as part of Hugging Face's Flax/JAX community week co-organized with Google's Flax, JAX, and Cloud teams (announcement).

KoBLIP

- 대규모 한국어 vision-language representation learning를 위한 computational resource 부족

THANK YOU

M o d e l i n g   T e a m **G**