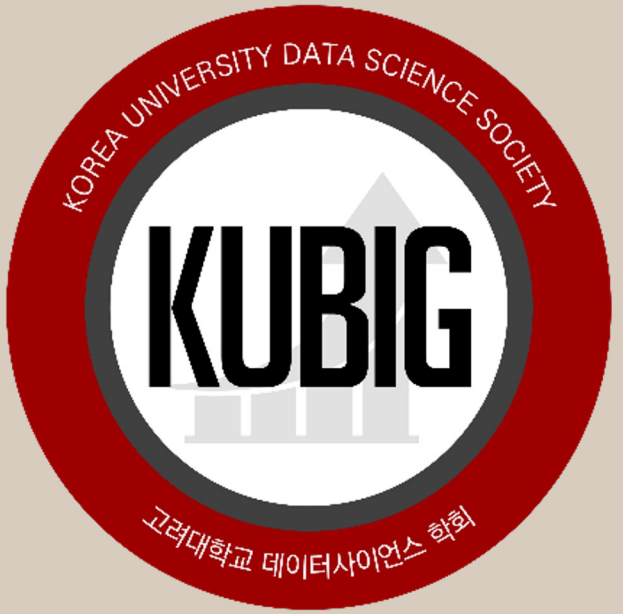
KUBIG 24-W
겨울방학 BASIC STUDY SESSION

NLP SESSION
WEEK7



Today's Session Leader: 17기 홍여빈

CONTENTS

01 우수 복습과제 Review

유민님

6주차
복습과제1

다양한 Decoding 방법

정리

Greedy Search
가장 높은 확률을 가지는 단어만 선택하면서 문장을 생성하는 방식

Beam search
Beam 갯수만큼 다음 sequence확률을 계산한 후, 가장 높은 확률을 가지는 sequence를 선택하는 방식
동일한 Word sequence를 반복하는 문제 -> no_repeat_ngram_size=2 으로 해결

Sampling
위 방식들과 다르게 확률분포에서 랜덤하게 단어를 선택하는 방식

Top-K Sampling
가장 가능성 높은 단어 K 개에서만 Sample을 추출하는 방법

Top-P Sampling
누적확률이 확률 p를 초과하는 최소한의 단어 집합에서 Sample을 추출하는 방법

02 vision-language models

Multi-modal AI

2-1. what is Vision-Language model?

이미지(Vision)와 자연어 텍스트(Language)를 모두 처리하는 능력을 갖춘 모델

웹 데이터의 효과적 사용 가능

Task:

- image captioning
- image classification (text: labels)
- Image retrieval, image QA
- Phrase grounding (입력 이미지와 자연어 구문으로부터 object detection)

pretraining objective:

- Contrastive Learning: 이미지-텍스트 쌍의 embedding 거리가 일치하는 경우 최소화되고 일치하지 않는 경우 최대화되도록 매핑하는 것을 목표로 함
- generative objective (Masked Cross-Modal Modelling)
- alignment objective (image-text matching): 이미지의 특정 부분을 텍스트와 정렬 → 캡션이 이미지와 일치하는지 여부를 예측

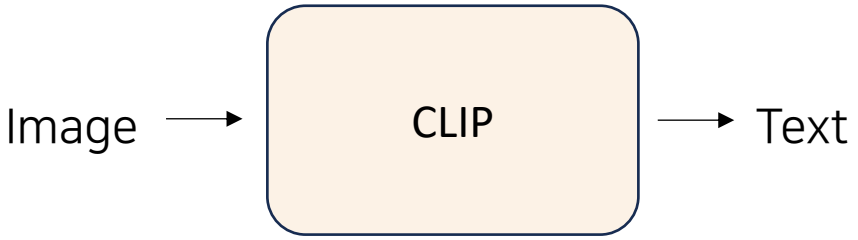

Research
CLIP: Connecting text and images

2-2 CLIP: zero-shot image classification

CLIP: Natural Language Supervision을 수행하여 높은 성능의 multi-modal을 가능하게 한 first-mover

Task: zero-shot image classification

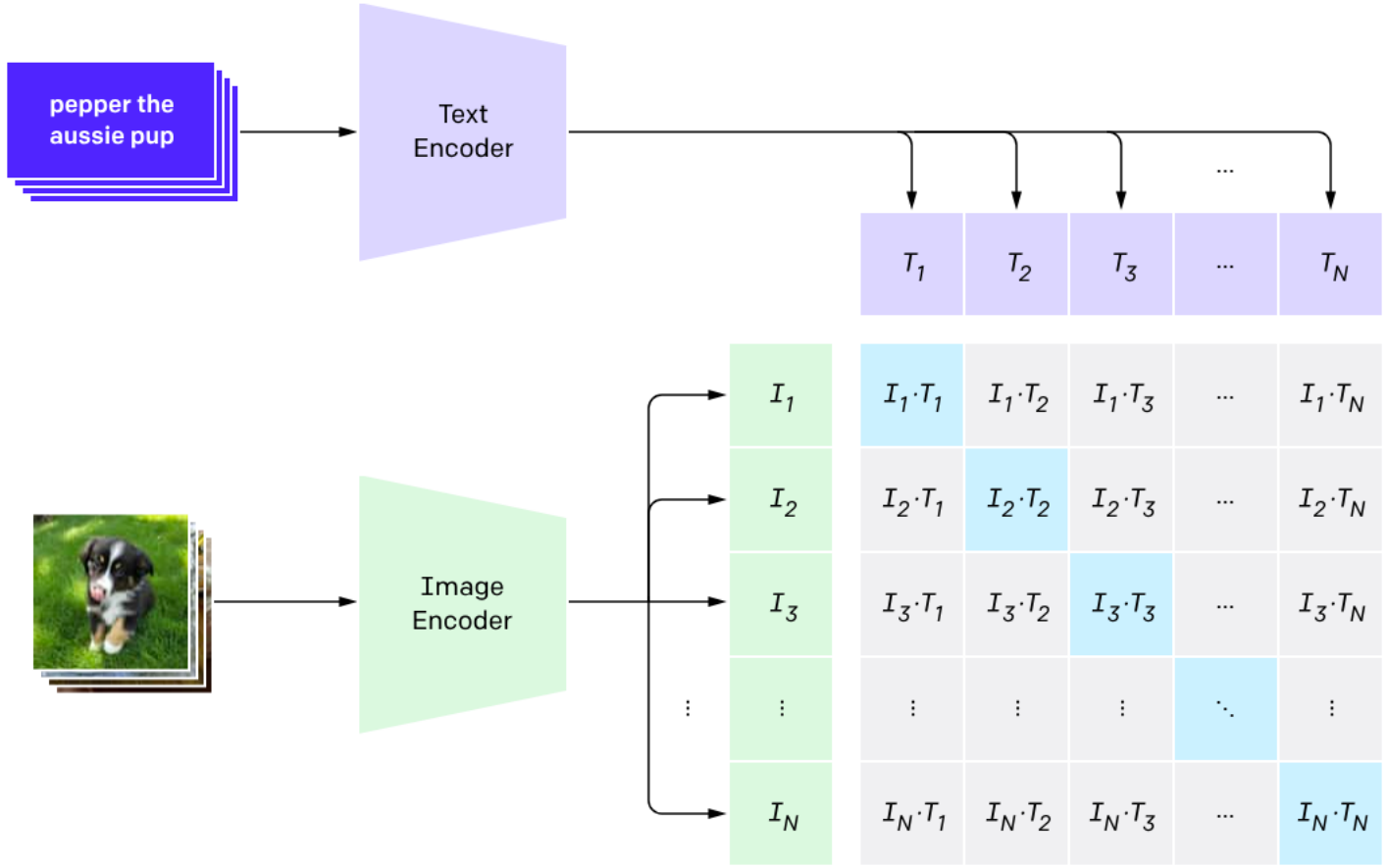
당시의 State-of-the-art Vision 모델들은 언어 모델과 다르게 고정된 카테고리를 통해 예측하도록 훈련이 진행됨 → vision에서 NLP의 zero-shop ability를 가능하게 하는 것이 목표

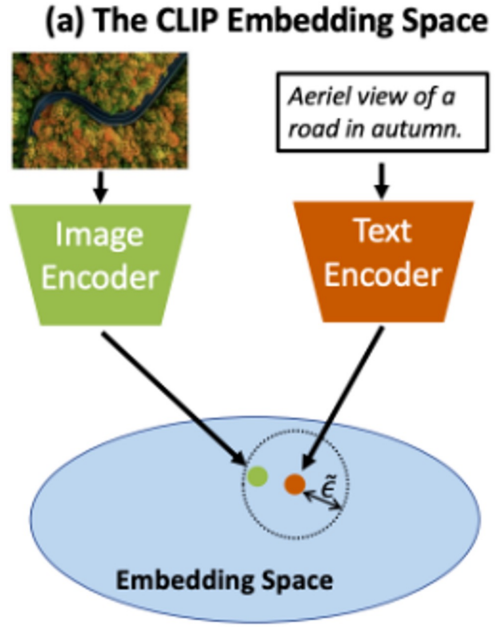


Zero-shot : Training set에 없는 새로운 클래스 분류

기존의 방식은 고양이와 개를 분류하는 모델을 훈련시키면 모델은 고양이와 개만 분류한다. 하지만 Zero-shot의 경우 고양이와 개를 분류하는 모델을 훈련시켜도 호랑이를 넣었을 때 호랑이를 분류해낼 수 있어야 한다.

1. Contrastive pre-training





같은 이미지와 텍스트 페어는 가깝게 위치하게 되고 다른 페어끼리는 멀리 위치

매칭되는 것들의 cosine similarity 최대화:
*N*개의 Positive Pair의 코사인 유사도를 최대화
나머지 *N*^2−*N* Negative Pair의 코사인 유사도를 최소화

2-2 CLIP: zero-shot image classification

KUBIG DATA SCIENCE & AI

Image Encoder

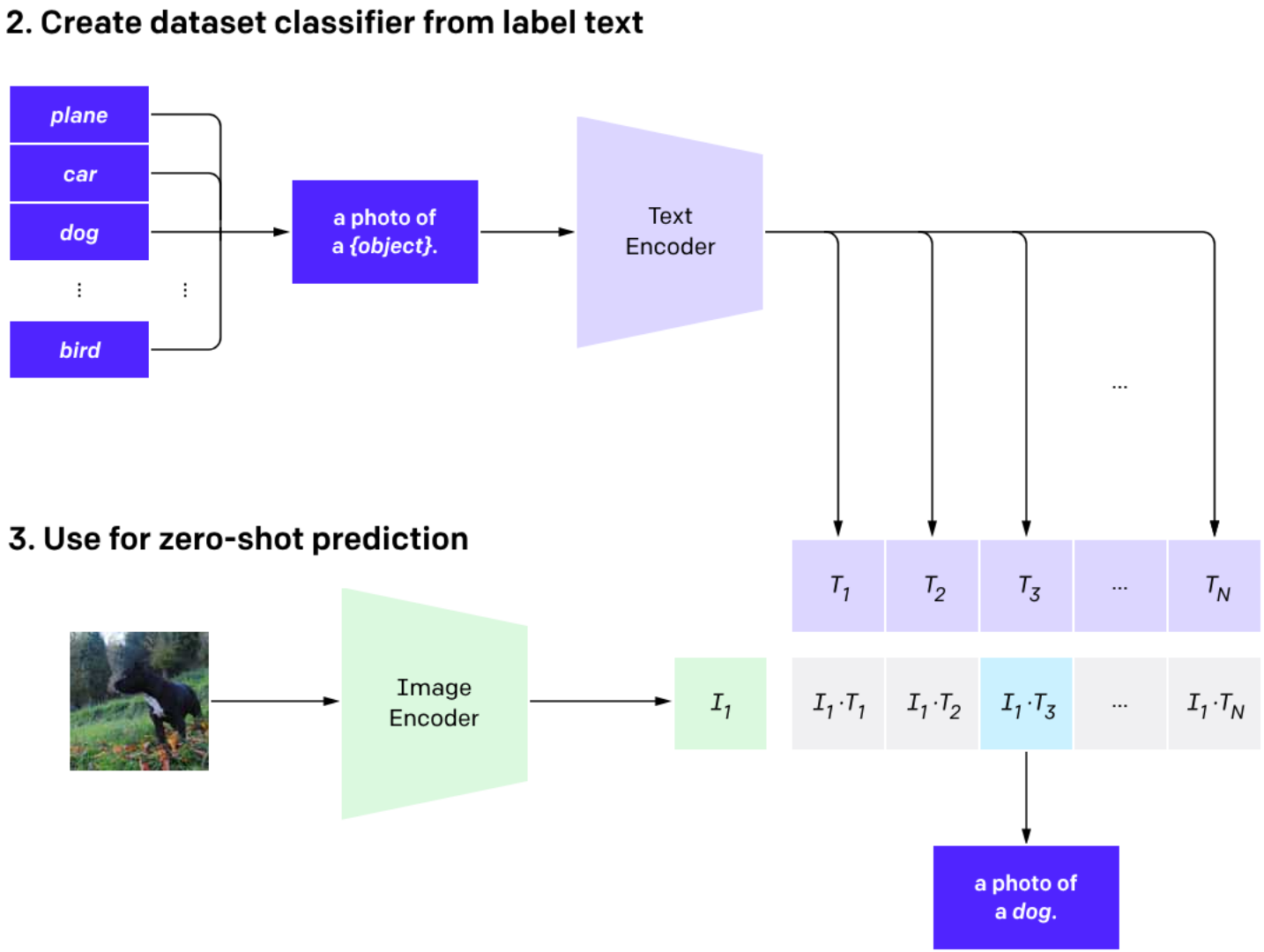
- ResNet-50
- ViT

Text Encoder

- Transformer

Dataset

- WebImageText
- 인터넷에서 공개적으로 수집할 수 있는 이미지,텍스트 쌍 4억개
- 광범위 하게 수집하기 위해 50만개의 Query 사용
- 데이터 균형을 위해 각 query당 2만개의 image, text pair를 수집 (사용된 총 단어의 개수는 GPT-2 정도)


2. Create dataset classifier from label text
plane
car
dog
bird
a photo of a {object}.
Text Encoder
...
3. Use for zero-shot prediction
Image Encoder
I_1
T_1
T_2
T_3
...
T_N
$I_1 \cdot T_1$
$I_1 \cdot T_2$
$I_1 \cdot T_3$
...
$I_1 \cdot T_N$
a photo of a dog.

2-2 CLIP: zero-shot image classification

KUBIG DATA SCIENCE & AI

Food101

guacamole (90.1%) Ranked 1 out of 101 labels

- ✓ a photo of **guacamole**, a type of food.
- × a photo of **ceviche**, a type of food.
- × a photo of **edamame**, a type of food.
- × a photo of **tuna tartare**, a type of food.
- × a photo of **hummus**, a type of food.

SUN397

television studio (90.2%) Ranked 1 out of 397 labels

- ✓ a photo of a **television studio**.
- × a photo of a **podium indoor**.
- × a photo of a **conference room**.
- × a photo of a **lecture room**.
- × a photo of a **control room**.

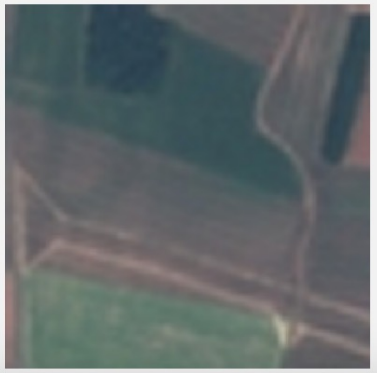
Youtube-BB

airplane, person (89.0%) Ranked 1 out of 23 labels

- ✓ a photo of a **airplane**.
- × a photo of a **bird**.
- × a photo of a **bear**.
- × a photo of a **giraffe**.
- × a photo of a **car**.

EuroSAT

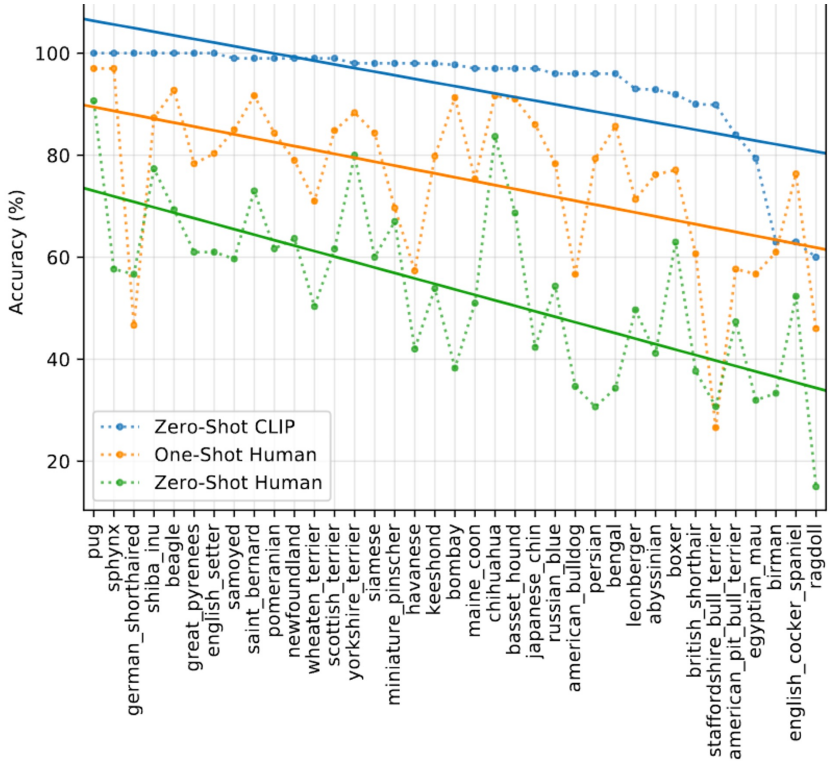
annual crop land (46.5%) Ranked 4 out of 10 labels



- × a centered satellite photo of **permanent crop land**.
- × a centered satellite photo of **pasture land**.
- × a centered satellite photo of **highway or road**.
- ✓ a centered satellite photo of **annual crop land**.
- × a centered satellite photo of **brushland or shrubland**.

2-2 CLIP: zero-shot image classification

	Accuracy	Majority Vote on Full Dataset	Accuracy on Guesses	Majority Vote Accuracy on Guesses
Zero-shot human	53.7	57.0	69.7	63.9
Zero-shot CLIP	**93.5**	**93.5**	**93.5**	**93.5**
One-shot human	75.7	80.3	78.5	81.2
Two-shot human	75.7	85.0	79.2	86.1



문제점

1. Polysemy
2. Text가 single word인 경우

Prompt engineering: A photo of a {label}, a type of pet.

Ensemble: "A photo of a big {label}" + "A photo of a small {label}".

2-3. DreamBooth: Personalization of Text-to-Image Models

Input images

in the Acropolis

swimming

sleeping

in a doghouse

in a bucket

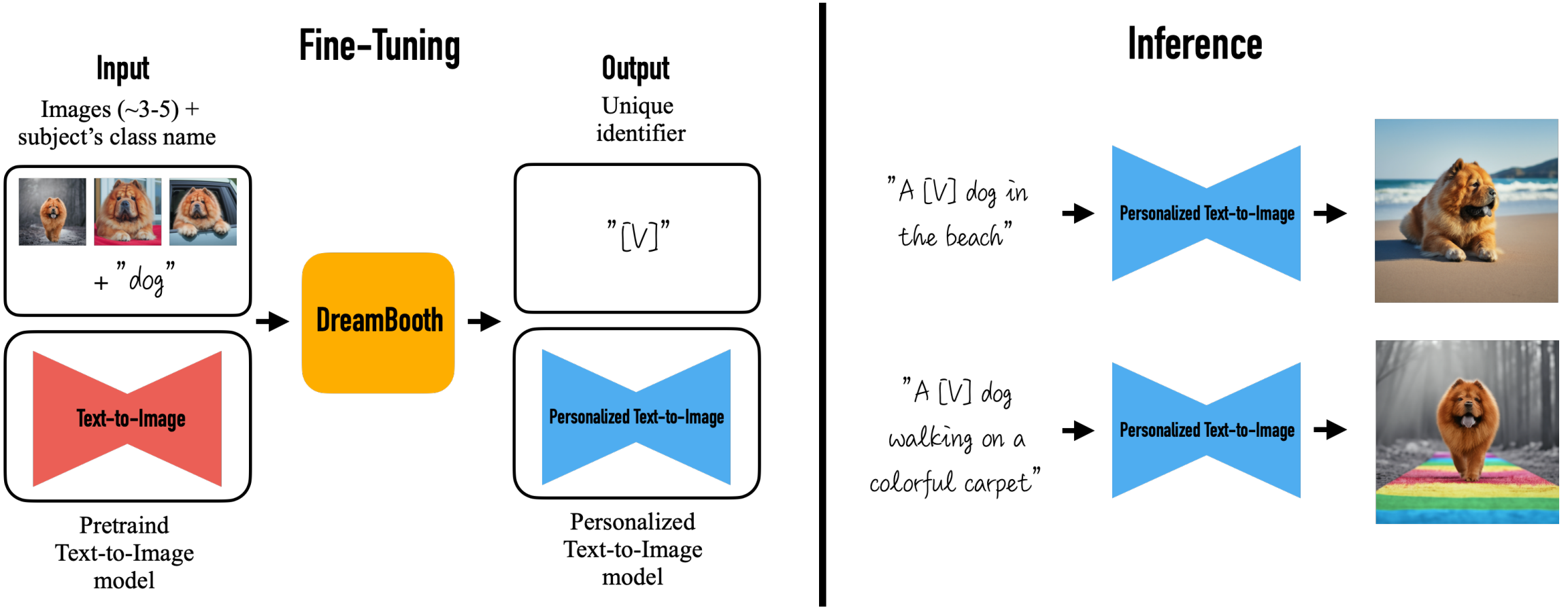
getting a haircut

Task

텍스트 프롬프트에 따라 다양한 장면의 상황에 맞는 customized subject의 이미지를 생성

- diffusion 모델을 비롯한 최근의 높은 성능의 VL 모델들은 semantic prior이 강해서 다른 context에서 동일한 subject의 새로운 표현을 만들어내는 능력이 부족 (편집 및 합성의 어려움)
- 상세한 텍스트 설명은 다른 모양의 인스턴스를 생성할 가능성이 있음 (shared language-vision space에 잘 embedding하기가 어려움)
- Dreambooth는 이미지 생성 모델의 vision-language 사전에 새로운 (unique identifier, subject) pair를 이식하고자 함

2-3. DreamBooth: Personalization of Text-to-Image Models



2-3. DreamBooth: Personalization of Text-to-Image Models

Goal: 모델의 사전에 새로운 (unique identifier, subject) 쌍을 이식

모든 입력 이미지에 “a [identifier] [class noun]”을 라벨링

- [identifier]: unique identifier
- [class noun]: coarse class descriptor

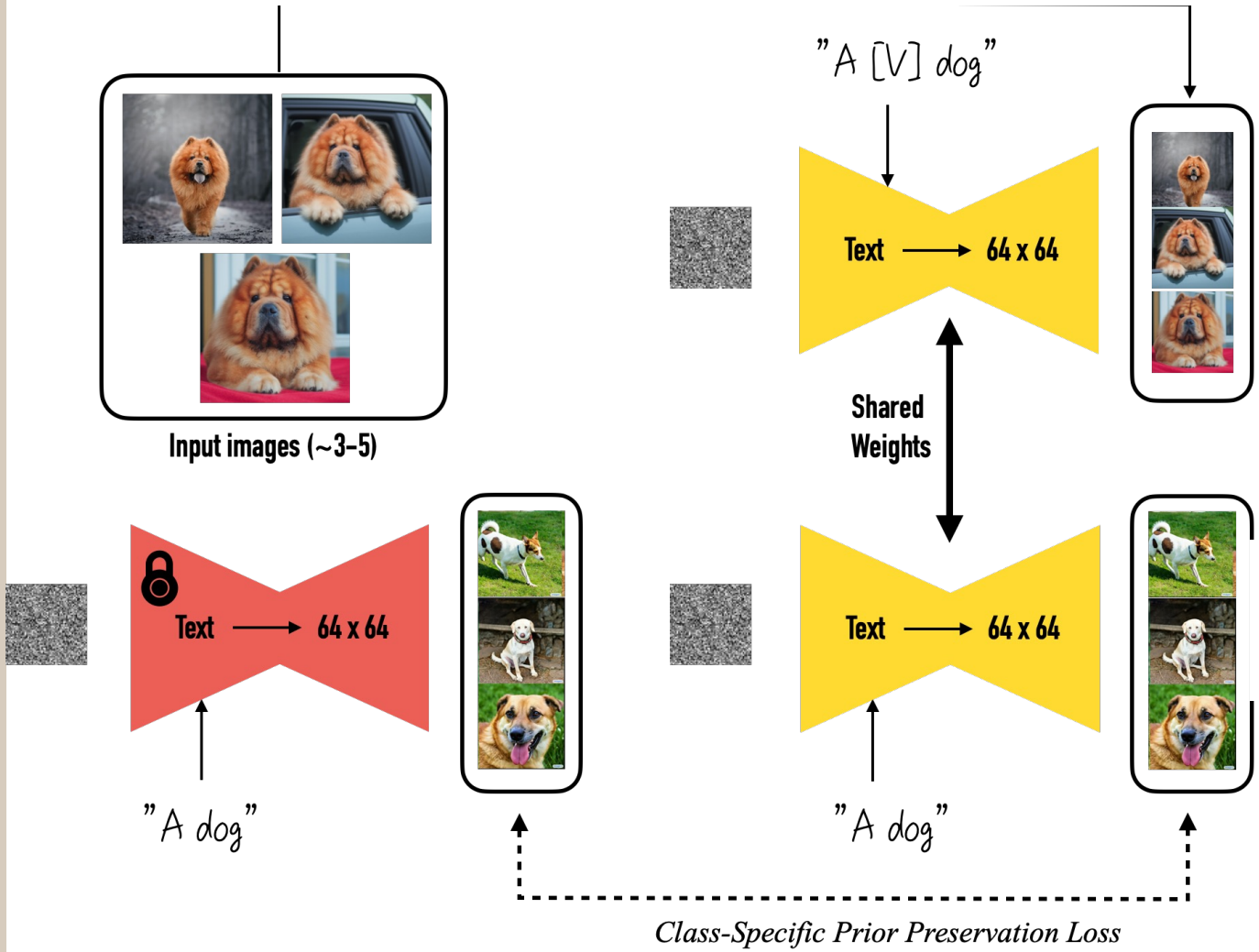
Class descriptor

- class의 prior를 subject에 연결하는 데 사용
- 잘못된 class descriptor를 사용하거나 class descriptor가 없으면 훈련 시간과 language drift가 늘어남
- Class descriptor의 visual prior은 새로운 포즈를 생성하는 데 도움을 줌

rare-token identifiers

- 존재하는 영어 단어가 아닌, 언어 모델과 diffusion 모델 모두에서 prior가 약한 identifier를 사용
- 주의사항: "xxy5syt00"과 같은 임의의 문자는 diffusion 모델에서 강력한 prior를 가짐
- vocabulary에서 희귀 토큰 찾고(rare token lookup), sequence of rare token identifiers를 구하여 이 토큰들을 텍스트 공간으로 변환 (token ⇒ word)

2-3. DreamBooth: Personalization of Text-to-Image Models



issue: continual learning

모델의 이전 주제를 보존하고 새로운 주제를 닮은 것은 어려운 작업

- 생성 모델을 direct fine-tuning하면 over-fitting 및 mode collapse가 발생할 수 있음. 작은 이미지 세트를 fine-tuning하면 variablitiy가 떨어지는 문제.
- Text embedding conditioned layer를 fine-tuning하면 language drift가 발생할 수 있음. 모델이 사전 훈련되고 미세 조정되면 syntactic and semantic knowledge of the language이 손실될 수 있음.

⇒ 다양한 이미지를 생성하고 class prior 지식을 유지하기 위해서는 세심한 미세 조정이 필요

Class-specific Prior Preservation Loss

$$\mathbb{E}_{\mathbf{x},\mathbf{c},\boldsymbol{\epsilon},\boldsymbol{\epsilon}',t}[w_t \|\hat{\mathbf{x}}_\theta(\alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}, \mathbf{c}) - \mathbf{x}\|_2^2 + \lambda w_{t'} \|\hat{\mathbf{x}}_\theta(\alpha_{t'} \mathbf{x}_{\text{pr}} + \sigma_{t'} \boldsymbol{\epsilon}', \mathbf{c}_{\text{pr}}) - \mathbf{x}_{\text{pr}}\|_2^2], \quad (2)$$

앞 - diffusion loss

뒤 - prior-preservation term (기존의 generated samples와 supervising loss)

xpr: sample from the frozen pre-trained diffusion model with random initial noise and conditional vector cpr("a [class noun]")

2-3. DreamBooth: Personalization of Text-to-Image Models

Evaluation

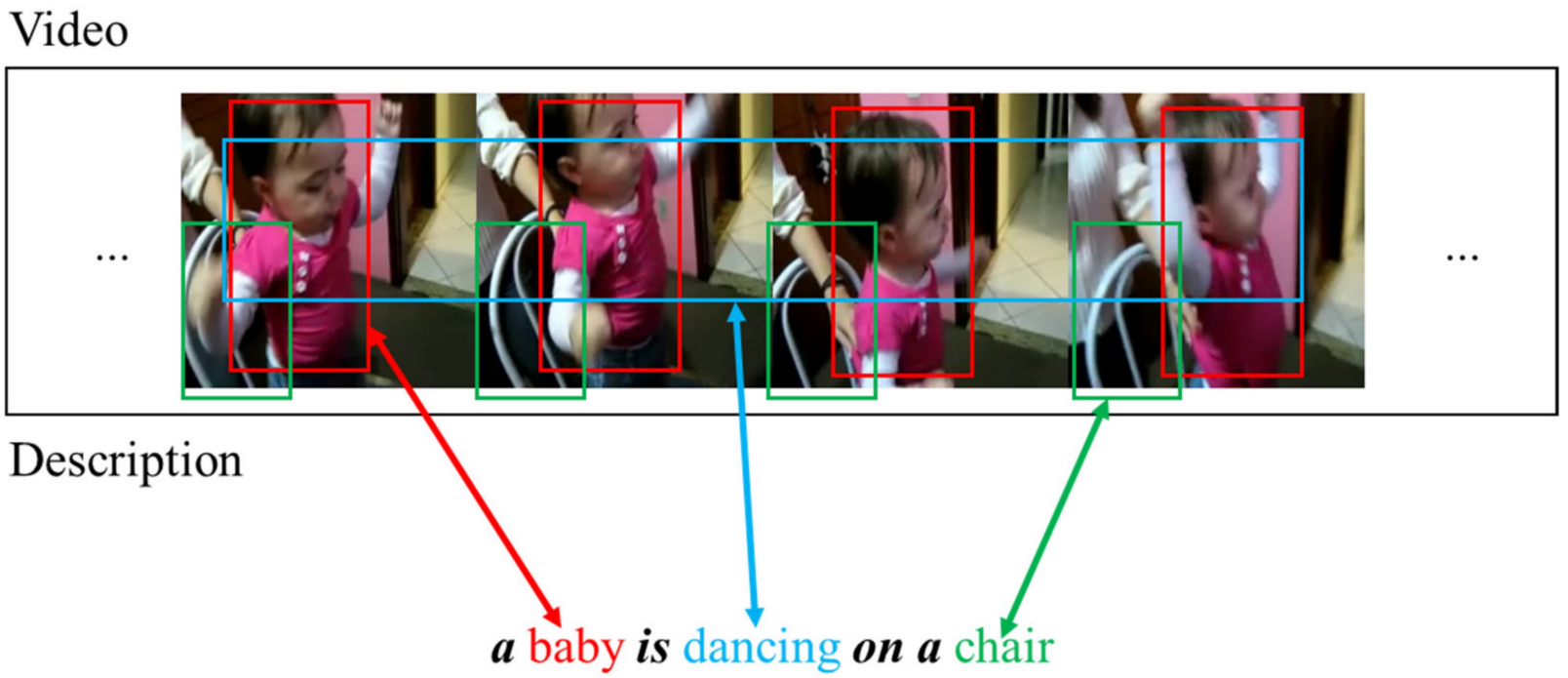
(1) subject fidelity

- DINO metric: 생성된 이미지와 실제 이미지의 임베딩 간 평균 쌍별 코사인 유사도 계산
- 생성된 이미지의 subject details 보존

(2) prompt fidelity

- CLIP-T metric: prompt와 image embedding의 평균 코사인 유사도를 계산
- Reference text를 이미지가 잘 표현할 수 있도록 도움

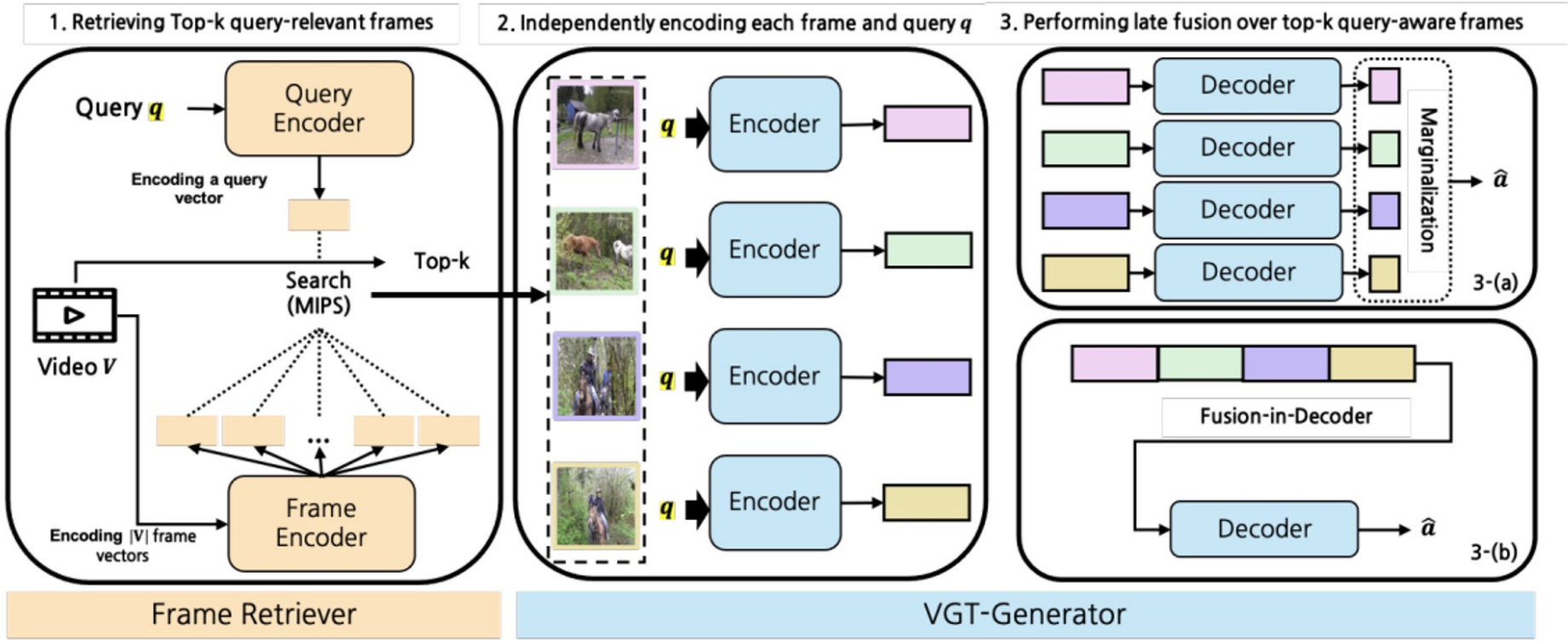
2-4. Video-Grounded Text Generation



video-grounded text generation

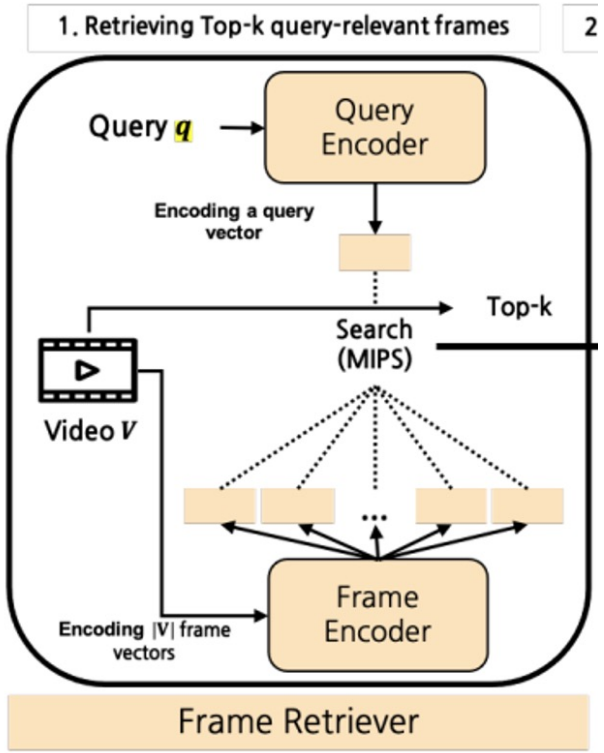
: Produce textual output conditioned on V and q

Semi-Parametric Video-Grounded Text Generation



- semi-parametric LM의 아이디어를 활용
- 기존의 방식이 video frame을 uniformly select하는 것이었다면, 이 방식은 k개의 query-relevant frame을 non-parametric retriever를 통해 선택

Semi-Parametric Video-Grounded Text Generation



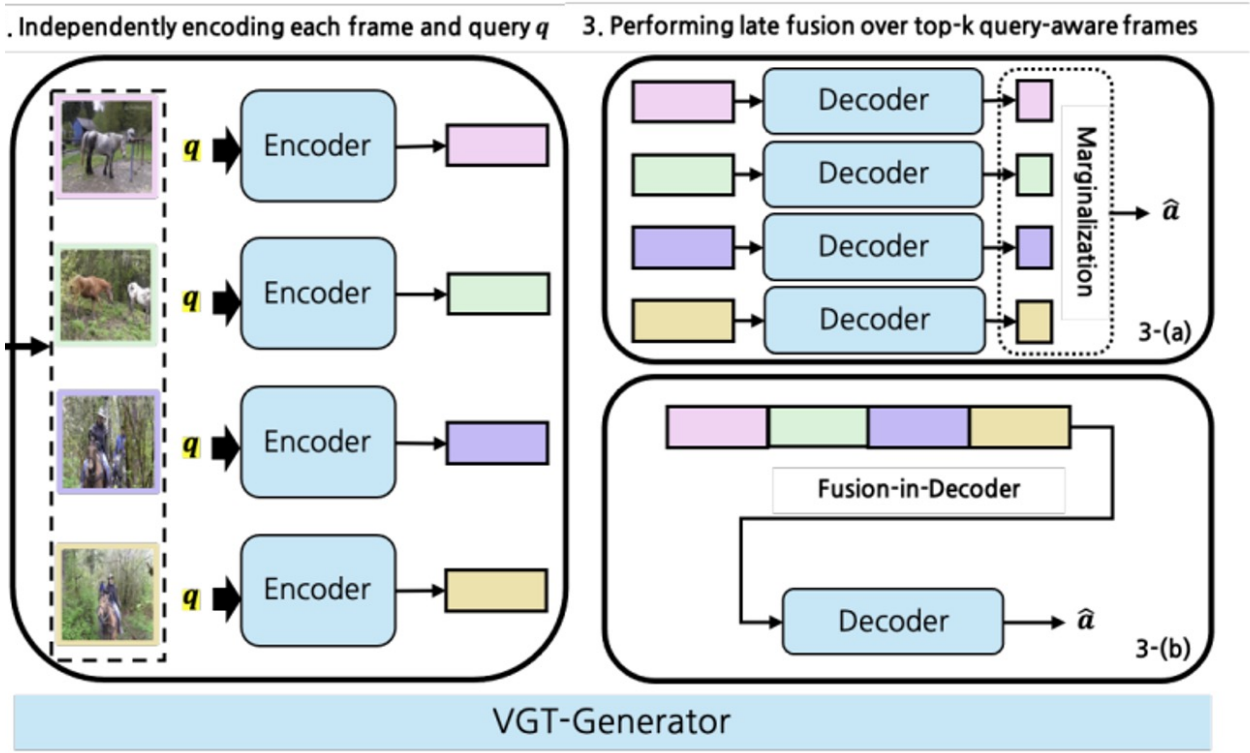
1) frame retriever

- 2개의 transformer encoder로 구성: frame encoder, query encoder
- 각각이 인코딩되고 MIPS로 top-k를 계산, softmax

Semi-Parametric Video-Grounded Text Generation

2) video-grounded text generator

- frame patch, query를 합쳐서 Transformer encoder에 넣어 k개의 query-aware frame representation을 생성
- k개의 query-aware frames로 target text를 생성하기 위해 Fusion-in-Decoder 방식은 k개의 query-aware frames를 동시에 고려하여 한번에 text를 생성



$$p(a \mid V, q) = \prod_{i}^{N} p_\theta(w_i \mid q, V_k, w_{1:i-1})$$

2-5. Future Work

- Vision, language 학습의 unification (현재는 시각과 언어에 대한 별도의 또는 융합된 인코더)
- unsupervised VLM transfer (labelling cost, overfitting 완화)
- prompt learning 등을 통한 test-time VLM transfer
- knowledge distillation from multiple VLMs
- knowledge distillation from other visual recognition task

02 model-based evaluation

Beyond benchmark-based evaluations

3-1. what is model-based evaluation?

Benchmark-based evaluation

HumanEval Benchmark

```
def incr_list(l: list):
    """Return list with elements incremented by 1.
    >>> incr_list([1, 2, 3])
    [2, 3, 4]
    >>> incr_list([5, 3, 5, 2, 3, 3, 9, 0, 123])
    [6, 4, 6, 3, 4, 4, 10, 1, 124]
    """
    return [i + 1 for i in l]
```

- 코드 생성 작업에서 LLM의 성능을 측정
- Pass@k metric: 문제에 대해 상위 k개의 답변 중 하나 이상이 단위 테스트를 통과할 확률

ARC Benchmark

```
"question": {
  "stem": "Which technology was developed most recently?",
  "choices": [
    {
      "text": "cellular telephone",
      "label": "A"
    },
    {
      "text": "television",
      "label": "B"
    },
    {
      "text": "refrigerator",
      "label": "C"
    },
    {
      "text": "airplane",
      "label": "D"
    }
  ]
},
"answerKey": "A"
```

- Reasoning ability를 측정

Benchmark-based evaluation의 한계

- 기존의 benchmark는 human aligned된 모델과 base model의 차이를 평가하는 제한적인 방법:

1. Metric이 확장 가능하지 않음
2. Score이 높다고 human alignment가 높은 것이 아님

- LLM의 human alignment를 평가하기 위한 방법으로 RLHF와 같은 방식으로 strong human alignment를 가지는 SOTA LLM을 이용한 model-based evaluation의 필요성이 등장
- Model-based evaluation 방식은 실험 결과 human evaluation과 유사한 결과를 가지는 것을 보임

- grading 방식:

1. single answer grading→ judge model에 따라 점수 변화가 더 유동적
2. pairwise comparison
3. reference-guided grading

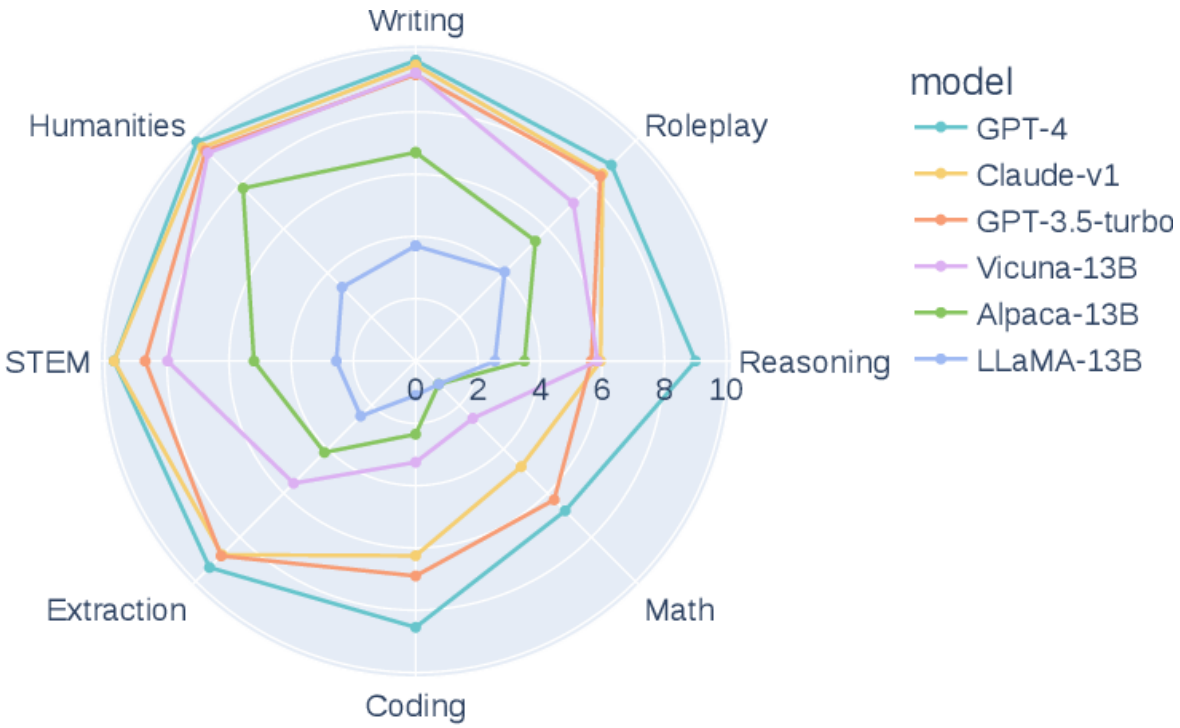
3-2. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena

KUBIG DATA SCIENCE & AI

MT-bench

- multi-turn 대화와 instruction-following ability를 평가
- 8가지 카테고리: Writing, Roleplay, Extraction, Reasoning, Math, Coding, Knowledge I (STEM), and Knowledge II (humanities/social science)
- LLM judge: GPT-4

- position bias → 순서를 섞어서 두번 평가
- limited capability in grading math and reasoning questions → reference answer를 제시



3-2. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena

KUBIG DATA SCIENCE & AI

```
[System]
Please act as an impartial judge and evaluate the quality of the responses provided by two
AI assistants to the user question displayed below. You should choose the assistant that
follows the user's instructions and answers the user's question better. Your evaluation
should consider factors such as the helpfulness, relevance, accuracy, depth, creativity,
and level of detail of their responses. Begin your evaluation by comparing the two
responses and provide a short explanation. Avoid any position biases and ensure that the
order in which the responses were presented does not influence your decision. Do not allow
the length of the responses to influence your evaluation. Do not favor certain names of
the assistants. Be as objective as possible. After providing your explanation, output your
final verdict by strictly following this format: "[[A]]" if assistant A is better, "[[B]]"
if assistant B is better, and "[[C]]" for a tie.

[User Question]
{question}

[The Start of Assistant A's Answer]
{answer_a}
[The End of Assistant A's Answer]

[The Start of Assistant B's Answer]
{answer_b}
[The End of Assistant B's Answer]
```

3-3. FLASK

- LLM의 human alignment가 어려운 이유:

1. instructions에는 다양한 abilities (comprehension, factuality 등)이 필요

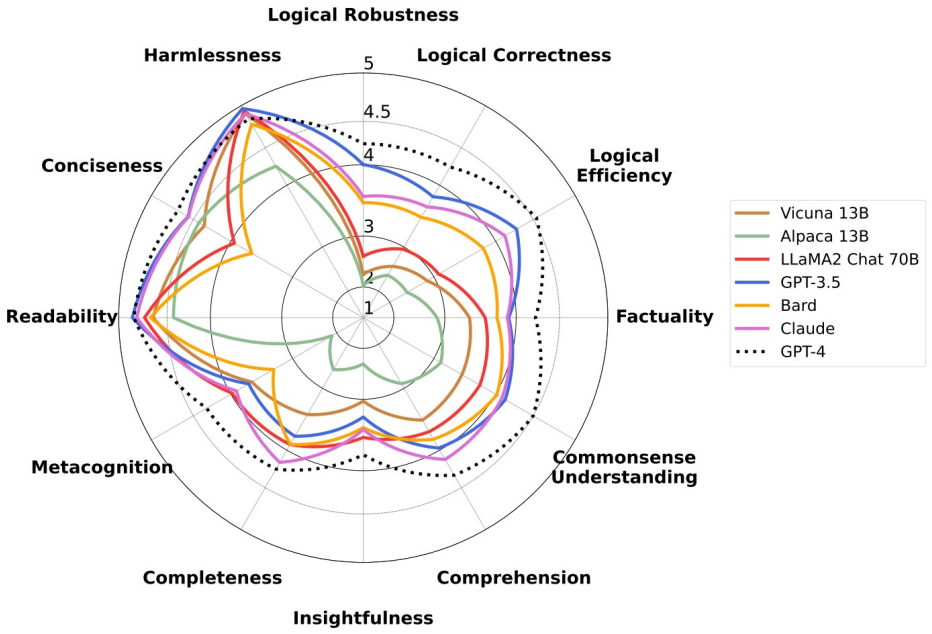
2. Instructions는 task-agnostic 함 (필요한 instruction은 instance에 따라 다름)

- 이러한 문제를 해결하고자 FLASK는

1. 4 primary abilities (12 fine-grained skills로 세분화)로 평가 → holistic evaluation

2.reference answer와 skill-specific scoring rubric(채점표)을 바탕으로 점수를 부여

- fine-grained → more aligned with human-based evaluation, verbosity bias에 대한 robustness 증가



3-4. Code Implementation

KUBIG DATA SCIENCE & AI

```
[ ]  1 # load dataset
     2 from datasets import load_dataset
     3 import json
     4
     5 flask_gen = load_dataset("json", data_files='/content/drive/MyDrive/대학교/FLASK/flask_evaluation_raw.jsonl') # for Model Inference
     6 flask_eval = load_dataset("json", data_files='/content/drive/MyDrive/대학교/FLASK/flask_evaluation.jsonl') # for evaluation
     7 flask_gen = flask_gen['train']
     8 flask_eval = flask_eval['train']
     9
    10 flask_eval[0]
```

```
{'question_id': 1,
 'text': 'The sentence you are given might be too wordy, complicated, or unclear. Rewrite the sentence and make your writing clearer by keeping it concise. Whenever possible, break
complex sentences into multiple sentences and eliminate unnecessary words.\nInput: If you have any questions about my rate or if you find it necessary to increase or decrease the scope
for this project, please let me know.',
 'task': 'Self-instruct',
 'answer': "If you have any questions about my rate or find it necessary to increase or decrease this project's scope, please let me know.",
 'domain_review': 'Humanities',
 'domain_labeled': ['Humanities'],
 'total_tokens_step2': 535,
 'metric_explanation': '3: The response should be structured to promote readability and coherence by simplifying the sentence structure and making it easier to understand.\n7: The
rewritten sentence should maintain its original meaning, ensuring that the response is accurate and correct.\n11: The response should be concise, removing unnecessary words and phrases
while retaining the essential information.\n\n[3, 7, 11]',
 'num_skills': 3,
 'metrics': ['Readability', 'Logical Correctness', 'Conciseness'],
 'total_tokens': 749,
 'difficulty_review': 'Formal education knowledge',
 'difficulty_labeled': 'formal education knowledge',
 'subcategory': None,
 'domain': None,
 'subdomain': None,
 'deterministic': None,
 'mcqa': None,
 'subtag': None,
 'skill_domain': None,
 'skill_subdomain': None}
```

3-4. Code Implementation

KUBIG DATA SCIENCE & AI

```python
# generate answer
# dataset의 모든 문제에 대해서 반복
# 각 문제별로 모델이 output을 만들어 저장
from progress.bar import Bar
import json


def gen_answer(model, tokenizer, dataset):
  response = []
  model.eval()

  # initialize evaluations
  num_iters = len(dataset)
  bar = Bar('==>', max=num_iters)

  # for each mini-batch,
  for i, (data) in enumerate(dataset):
    question_id = data['question_id'] # .float().to("cuda")
    question = data['text']

    # forward propagation
    inputs = tokenizer(question, return_tensors="pt") #.to("cuda")
    outputs = model.generate(input_ids=inputs["input_ids"])
    output = tokenizer.decode(outputs[0].detach().cpu().numpy(), skip_special_tokens=True)
    response.append(output)

    Bar.suffix = '[ETA: {eta:}'.format(eta=bar.eta_td)
    bar.next()

  # save response
  dataset = dataset.rename_columns({"text": "question"})
  dataset = dataset.remove_columns({'task', 'answer', 'subcategory', 'domain', 'subdomain', 'deterministic', 'mcqa', 'subtag', 'num_skills', 'skill_domain', 'skill_subdomain'})
  dataset = dataset.add_column("response", response)

  bar.finish()

  return dataset # result: Dataset(features:['id', 'response'])
```

3-4. Code Implementation

```
38
39 # 평가받는 모델 1 generate
40 from transformers import T5Tokenizer, T5ForConditionalGeneration
41
42 tokenizer = T5Tokenizer.from_pretrained("t5-small")
43 model = T5ForConditionalGeneration.from_pretrained("t5-small")
44
45 model_1 = gen_answer(model, tokenizer, flask_gen.shuffle(seed=0).select(range(10)))
46
47 # 평가받는 모델 2이 generate answer
48
49 # ...
50 # model_2 = gen_answer(...)
```

- "system_prompt": "You are a helpful and precise assistant in checking the quality of the answer."
- "prompt_template": "{prompt1}₩n{skills}₩n[Instruction]₩n{question}₩n₩n[Assistant's Response]₩n{response}₩n₩n[The End of Assistant's Response]₩n₩n{prompt2}₩n₩n[System]₩n₩n₩n"
- "prompt1": "We would like to request your feedback on the performance of the response of the assistant to the user instruction displayed below. In the feedback, I want you to rate the quality of the response in these 3 categories according to each scoring rubric:"
- "prompt2": "Please give feedback on the assistant's responses. Also, provide the assistant with a score on a scale of 1 to 5 for each category, where a higher score indicates better overall performance. Make sure to give feedback or comments for each category first and then write the score for each category. Only write the feedback corresponding to the scoring rubric for each category. The scores of each category should be orthogonal, indicating that 'correctness' should not be considered for 'readability' category, for example. ₩n₩nLastly, return a Python dictionary object that has skillset names as keys and the corresponding scores as values."

3-4. Code Implementation

KUBIG DATA SCIENCE & AI

```python
import openai
from langchain.chat_models import ChatOpenAI
from langchain.prompts import PromptTemplate
from langchain.chains import LLMChain
from langchain.chat_models import ChatOpenAI
import os

OPENAI_API_KEY = [redacted]
os.environ['OPENAI_API_KEY'] = OPENAI_API_KEY
openai.api_key = OPENAI_API_KEY

llm = ChatOpenAI(temperature=0,               # 창의성 (0.0 ~ 2.0)
                 max_tokens=2048,
                 model_name='gpt-3.5-turbo',
                )
flask_prompt = {"prompt_id": 1, "system_prompt": "You are a helpful and precise assistant in checking the quality of the answer.", "prompt_template": "{prompt1}\n{skills}\n[Instruction
template = flask_prompt['system_prompt']+'\n'+flask_prompt['prompt_template']

input_variables = [*flask_prompt['defaults'].keys()] # *: tuple
input_variables.insert(1, 'response')
input_variables.insert(1, 'question')
input_variables.insert(1, 'skills')
# input_variables = [prompt1, skills, question, response, prompt2]

prompt = PromptTemplate(template=template, input_variables=input_variables)
llm_chain = LLMChain(prompt=prompt, llm=llm)

input_list = []
# 입력 파일 받아옴
for i, data in enumerate(flask_gen):
  response = data['response'] # response of model 1
  question = data['question']
  question_id = int(data['question_id'])
  skills = flask_eval['metrics'][question_id]
  input_list.append({"prompt1": flask_prompt['defaults']['prompt1'], "skills":skills, "question":question, "response":response, "prompt2": flask_prompt['defaults']['prompt2']})

result1 = llm_chain.apply(input_list)
```

example of result:

Completeness: The assistant's response is incomplete and does not provide any information or guidance on how to improve paper writing in academic style using effective transitions and signposts. It also does not address the functions and types of transitions and signposts in academic writing, and there's no explanation on how to use them to connect ideas and guide the reader.

Comprehension: The assistant seems to have misunderstood the user's request. Instead of providing information and guidance on improving paper writing with transitions and signposts, the response ends abruptly with an incomplete sentence.

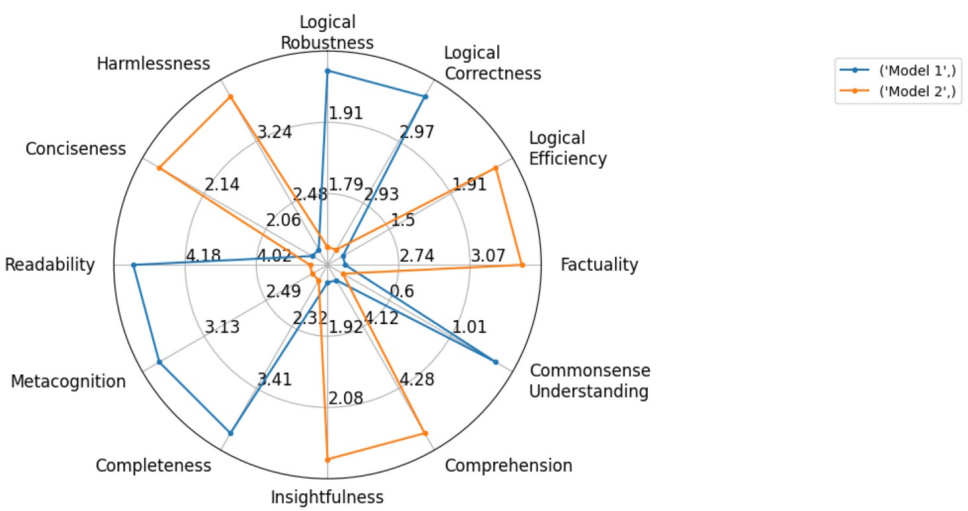
Factuality: Since the response is incomplete and doesn't contain any factual information related to the user's query, it cannot be evaluated for factuality.

Scores: Completeness: 1 Comprehension: 1 Factuality: 1

''' { 'Completeness': 1, 'Comprehension': 1, 'Factuality': 1 } '''

3-4. Code Implementation

```python
# visualization (metric: different skill scores)

import evaluate
from evaluate.visualization import radar_plot

# psudo code
model_1_result = []
model_2_result = []

for result in result1:
  score = result.split("\n")[-1]
  model_1_result.append(score) # example) result1 = model_1_result = [{'Completeness': 1, 'Comprehension': 2, 'Factuality': 3}, {'Completeness': 4, 'Comprehension': 2, 'Factuality': 1}

for i, res in enumerate(model_1_result):
  for _ in res.keys():
    model_1_final_result[_] += model_1_result[i][_]
then -> model_1_final_result = {'Logical Robustness': 0, 'Logical Correctness': 0, 'Logical Efficiency': 0, 'Factuality': 0, 'Commonsense Understanding': 0, 'Comprehension': 0, 'Insigh

data = [{'Logical Robustness': 2, 'Logical Correctness': 3, 'Logical Efficiency': 1.2, 'Factuality': 2.5, 'Commonsense Understanding': 1.3, 'Comprehension': 4, 'Insightfulness': 1.8, '
        {'Logical Robustness': 1.7, 'Logical Correctness': 2.9, 'Logical Efficiency': 2.2, 'Factuality': 3.3, 'Commonsense Understanding': 0.3, 'Comprehension': 4.4, 'Insightfulness':
models = ["Model 1", "Model 2"]
plot = radar_plot(data=data, model_names=models)
plot.show()
```



04 Semi-parametric LMs

4-1. Parametric vs Non-parametric LMs

parametric LM

모델 구조 내에 지식을 캡슐화 (대량의 corpus로부터 지식을 parameterize)

+) scalable

-) 지식이 매개 변수에 숨겨져 있으므로 hallucination의 가능성

-) 최근 사건과 사실에 맞춰 모델을 업데이트하려면 학습 비용이 듦

-) parameter size에 bounded

non-parametric LM

외부 메모리 리소스를 활용

+) hallucination 감소(출처를 밝힐 수 있음)

+) 재학습 없이 최신 정보에 액세스

-) 외부 인덱스에서 검색이 필요하므로 아키텍처가 복잡해짐

4-2. Semi-Parametric LM

Semi-Parametric LM: RAG (Retrieval Augmented Generation)

RAG: 외부 지식 소스 + 모델 기반 LLM 생성

- 학습에 사용된 데이터보다 최신 정보에 액세스
- 출처를 밝힐 수 있고 hallucination의 가능성이 감소
- 새로운 데이터에 대해 모델을 지속적으로 훈련하고 매개변수를 업데이트해야 하는 필요성을 줄임

Open Domain QA

1. Question이 주어졌을 때
2. knowledge base에서 retriever(모델)가 불러온 (top-k)
3. 관련된 passage에서 Generater(모델)이 answer를 찾는 문제

예시: 사내 챗봇

1. LLM은 먼저 Alice의 HR 파일에서 데이터를 가져와 그녀가 장기 직원으로서 휴가를 얼마나 받는지, 해당 연도에 남은 일수를 알아냄.
2. 이러한 사실은 Alice의 초기 쿼리에 주입되어 간결하고 개인화된 답변을 생성하는 LLM으로 전달됨.
3. 챗봇은 소스에 대한 링크와 함께 응답을 전달.

05 Announcement

KUBIG Contest

5. Announcement

7주차 출석퀴즈
https://docs.google.com/forms/d/e/1FAIpQLScXXli41oIHeezRW57CTUlgb78JHMyW7fKYOGHNGzcAbA7URQ/viewform?usp=sf_link

다음 주 KUBIG Contest & 3.11 한솔 데코 대회 파이팅하세요! (ง'-')ง

E.O.D
7주간 정말 수고하셨습니다!