

Experimental data-analysis

재현 수치의 정당성 확보 (관련 연구 조사를 바탕으로)

학생 원종찬, 최재원

Prof. Jae-Hong Lee, Jun-Hyung Park

- 250922
 - SAKURA 및 JASCO 논문 발표
- 250929
 - SALMONN에 대해 후처리를 이유로 재현상 어려움을 제시
- 251013
 - SAKURA 및 SALMONN의 재현 결과 공유

identical 하게 같은 수치 나오는 경우 드뭄 : model의 평가 수치구현에 대한 신뢰도에 의심이 있는 상황

1. SALMONN을 벤치마크 이상으로 인용한 연구 (논문) 20편 조사 (우선 20편)
2. 사용 task data가 SALMONN 7p table2와 같은 data사용 여부 확인 (같은 데이터여야,,)
3. 해당 논문의 레포로 코드를 통해 어떤 조건으로 재현했는지를 확인 (혹은 논문에서 재현 방법을 제시했다면 확인)

AIR-Bench: Benchmarking Large Audio-Language Models — 대형 audio-LLM 평가용 벤치로 SALMONN을 포함해 여러 모델을 비교·점수 수록.

Qwen2-Audio (technical report / arXiv, 2024) — Qwen-Audio 비교표에 SALMONN 점수를 넣어 광범위 비교(음성·음향·뮤직 태스크).

AudioBench: A Universal Benchmark for Audio Large Models (NAACL 2025) — 여러 audio-LLM(=SALMONN 포함) 성능 비교.

MuChoMusic: Evaluating Music Understanding in Multimodal Audio-Language Models (ISMIR / MuChoMusic) — 음악 이해 벤치에서 SALMONN을 평가대상으로 포함(음악 캡셔닝/MC 관련 표).

MMAU: A Massive Multi-Task Audio Understanding benchmark (ICLR 2025 spotlight) — 멀티태스크 오디오 평가에서 SALMONN을 비교 대상에 포함.

FunAudioLLM (Tongyi SpeechTeam technical report) — 대형 오디오 모델 소개/비교 보고서로 SALMONN 수치 인용.

VoxLM: Unified decoder-only models (ICASSP 2024 / related work) — 음성-LLM 비교 문헌에서 SALMONN을 baseline으로 언급·수치 비교.

video-SALMONN (audio-visual LLM; G. Sun et al.) — SALMONN 계열의 AV 확장으로 원작·점수 비교 문맥에서 SALMONN을 핵심 인용.

SALMon: A suite for acoustic language model evaluation (ICASSP 2025) — 새로운 SALMon 평가 스위트 논문(벤치마크)에서 SALMONN을 비교·인용.

Towards Holistic Evaluation of Large Audio-Language Models — 평가 논문으로 SALMONN을 비교·인용.

Are you really listening? Boosting Perceptual Awareness in Audio LLM — audio-LLM 개선 연구에서 SALMONN을 비교 baseline으로 다수 지표 인용.

Investigating Modality Contribution in Audio LLMs for Music — 음악 태스크 성능 분석에서 SALMONN 점수 비교.

Leveraging Content and Acoustic Representations for ... — LLM 기반 음성/감정 관련 연구에서 SALMONN 수치·설정 인용.

AudioBench / AIR-Bench code repositories — 벤치마크 코드·리포트에서 SALMONN을 포함한 수치

SALMONN family / video-SALMONN repos — SALMONN 변형·후속(AV 버전 포함) 레포·데모 문서에서 원본 SALMONN 점수/비교를 인용·정리

Audio Flamingo 2 (NVIDIA/ADLR report) — Audio Flamingo 계열 평가표에서 SALMONN을 포함한 여러 모델과 성능 비교.

DiVA : Distilling an End-to-End Voice Assistant (ACL 2025 / tech report) — 음성 어시스턴트 구축 논문으로 SALMONN을 비교 baseline으로 사용.

Typhoon-Audio (Interspeech 2025) — SALMONN 아키텍처·데이터/평가 방식을 참고·비교하며 성능 인용.

On the Landscape of Spoken Language Models (survey / arXiv Apr 2025) — 스펙트럼 정리 논문으로 SALMONN을 대표적 사례·비교표에 포함.

Recent Advances in Speech Language Models: A Survey (ACL 2025 long) — ACL 서베이에서 SALMONN을 주요 선행으로 정리·점수 인용.

AIR-Bench: Benchmarking Large Audio-Language Models via Generative Comprehension

Qian Yang^{1*†}, Jin Xu^{2*}, Wenrui Liu¹, Yunfei Chu², Ziyue Jiang¹, Xiaohuan Zhou²

Yichong Leng², Yuanjun Lv², Zhou Zhao^{1‡}, Chang Zhou^{2‡}, Jingren Zhou²

¹Zhejiang University, ²Alibaba Group

{qyang1021, liuwenrui, ziyuejiang, zhaozhou}@zju.edu.cn

{renjun.xj, fay.cyf, shiyi.zxh, lengyichong.lyc, lvyuanjun.lyj}@alibaba-inc.com

{ericzhou.zc, jingren.zhou}@alibaba-inc.com

| Types | Task | Dataset-Source | Num |
|--------|----------------------------------|--------------------------------------|------|
| Speech | Speech grounding | Librispeech (Panayotov et al., 2015) | 0.9k |
| | Spoken language identification | Covost2 (Wang et al., 2020b) | 1k |
| | Speaker gender recognition | Common voice (Ardila et al., 2019) | 1k |
| | (biologically) | MELD (Poria et al., 2018) | |
| | Emotion recognition | IEMOCAP (Busso et al., 2008) | 1k |
| | | MELD (Poria et al., 2018) | |
| | Speaker age prediction | Common voice (Ardila et al., 2019) | 1k |
| | Speech entity recognition | SLURP (Bastianelli et al., 2020) | 1k |
| | Intent classification | SLURP (Bastianelli et al., 2020) | 1k |
| | Speaker number verification | VoxCeleb1 (Nagrani et al., 2020) | 1k |
| Sound | Synthesized voice detection | FoR (Reimao and Tzerpos, 2019) | 1k |
| | Audio grounding | AudioGrounding (Xu et al., 2021) | 0.9k |
| | Vocal sound classification | VocalSound (Gong et al., 2022) | 1k |
| | Acoustic scene classification | CochlScene (Jeong and Park, 2022) | 1k |
| | | TUT2017 (Mesaros et al., 2017) | |
| Music | Sound question answering | Clotho-AQA (Lipping et al., 2022) | 1k |
| | | AVQA (Yang et al., 2022) | |
| | Music instruments classification | Nsynth (Engel et al., 2017) | 1k |
| | | MTJ-Jamendo (Bogdanov et al., 2019) | |
| | Music genre classification | FMA (Defferrard et al., 2016) | 1k |
| | | MTJ-Jamendo (Bogdanov et al., 2019) | |
| | Music note analysis-pitch | Nsynth (Engel et al., 2017) | 1k |
| | Music note analysis-velocity | Nsynth (Engel et al., 2017) | 1k |
| | Music question answering | MUSIC-AVQA (Li et al., 2022) | 0.8k |
| | Music emotion detection | MTJ-Jamendo (Bogdanov et al., 2019) | 1k |

Table 1: The statistics of the foundation benchmark.

| Categories | Qwen-Audio | Qwen-Audio Turbo | SALMONN | BLSP | NExT-GPT | SpeechGPT | PandaGPT | Whisper+GPT-4 |
|----------------------------------|------------|------------------|---------|-------|----------|-----------|----------|---------------|
| Speech grounding | 56.1% | 45.4% | 25.3% | 25.0% | 25.4% | 28.8% | 23.0% | 35.0% |
| Spoken language identification | 92.8% | 95.9% | 28.1% | 30.8% | 23.7% | 39.6% | 34.6% | 96.8% |
| Speaker gender recognition | 67.2% | 82.5% | 35.5% | 33.2% | 57.0% | 29.2% | 66.5% | 21.9% |
| Emotion recognition | 43.2% | 60.0% | 29.9% | 27.4% | 25.7% | 37.6% | 26.0% | 59.5% |
| Speaker age prediction | 36.0% | 58.8% | 48.7% | 51.2% | 62.4% | 20.4% | 42.5% | 41.1% |
| Speech entity recognition | 71.2% | 48.1% | 51.7% | 37.2% | 26.1% | 35.9% | 34.0% | 69.8% |
| Intent classification | 77.8% | 56.4% | 36.7% | 46.6% | 25.6% | 45.8% | 28.5% | 87.7% |
| Speaker number verification | 35.3% | 54.3% | 34.3% | 28.1% | 25.4% | 32.6% | 43.2% | 30.0% |
| Synthesized voice detection | 48.3% | 69.3% | 50.0% | 50.0% | 30.8% | 39.2% | 53.1% | 40.5% |
| Audio grounding | 23.9% | 41.6% | 24.0% | 34.6% | 62.2% | 26.1% | 38.3% | / |
| Vocal sound classification | 84.9% | 78.1% | 45.3% | 29.8% | 23.5% | 26.2% | 31.6% | / |
| Acoustic scene classification | 67.5% | 61.3% | 34.1% | 25.2% | 24.1% | 23.7% | 55.7% | / |
| Sound question answering | 64.6% | 62.8% | 28.4% | 36.1% | 18.8% | 33.9% | 48.7% | / |
| Music instruments classification | 59.1% | 59.6% | 41.3% | 22.8% | 24.3% | 29.1% | 47.7% | / |
| Music genre classification | 71.2% | 77.1% | 45.3% | 26.1% | 28.1% | 29.3% | 39.8% | / |
| Music note analysis-pitch | 28.6% | 30.1% | 26.4% | 23.5% | 25.1% | 24.1% | 26.4% | / |
| Music note analysis-velocity | 25.4% | 25.1% | 22.8% | 24.9% | 23.1% | 25.2% | 27.2% | / |
| Music question answering | 48.2% | 62.5% | 54.6% | 31.0% | 47.1% | 31.3% | 50.7% | / |
| Music emotion detection | 36.1% | 39.0% | 32.2% | 28.3% | 25.4% | 29.7% | 36.7% | / |

Table 6: The accuracy of each model across all tasks in the foundation benchmark.

- <https://arxiv.org/abs/2506.12285v1>
 - CMI-Bench: A Comprehensive Benchmark for Evaluating Music Instruction Following
 - BLEU4: wordpunc_tokenize로 토큰화해 계산
 - WER: typical patterns 삭제 후 계산

```
1 def compute_wer_cer(prediction, reference):
2     # Clean the prediction (remove prefix)
3     patterns = [
4         r".*? lyrics .*?are.*?:",
5         r".*? content .*?is.*?:",
6         r".*? transcription .*?is.*?:",
7         r".*? text .*?is.*?:",
8         "<s>",
9         "</s>"
10    ]
11    for pattern in patterns:
12        prediction = re.sub(pattern, '', prediction).strip()
13
14    def clean_string(text):
15        # text = text.translate(str.maketrans('', '', string.punctuation))
16        text = text.translate(str.maketrans('', '', '!"#$%&()*~`^_{}|;:\'"/>
17        text = text.lower().replace("\n", " ")
18        text = convert_digits_to_words(text.split())
19        text = " ".join(text)
20        return text
21    prediction = clean_string(prediction)
22    reference = clean_string(reference)
23
24    # Compute WER and CER using jiwer
25    wer = jiwer.wer(reference, prediction)
26    cer = jiwer.cer(reference, prediction)
27
28    return wer, cer
```

Lyrics Transcription. We extract lyrics from model outputs by removing typical prefixes (e.g., "lyrics is as follows:"). Word Error Rate (WER) and Character Error Rate (CER) are computed against ground-truth lyrics.

```
1 class WordPunctTokenizer(RegexTokenizer):
2     """
3     Tokenize a text into a sequence of alphabetic and
4     non-alphabetic characters, using the regexp ``\w+|[\^\w\s]+``.
5
6     >>> from nltk.tokenize import WordPunctTokenizer
7     >>> s = "Good muffins cost $3.88\nin New York. Please buy me\ntwo of them.\n\nThanks."
8     >>> WordPunctTokenizer().tokenize(s) # doctest: +NORMALIZE_WHITESPACE
9     ['Good', 'muffins', 'cost', '$', '3', '.', '88', 'in', 'New', 'York',
10     '.', 'Please', 'buy', 'me', 'two', 'of', 'them', '.', 'Thanks', '.']
11     """
12
13     def __init__(self):
14         RegexTokenizer.__init__(self, r"\w+|[\^\w\s]+")
```

```
1 def music_captioning(result_list):
2     scorer = rouge_scorer.RougeScorer(['rougeL'], use_stemmer=True)
3     rouge_score, bleu_score, meteor_score = 0, 0, 0, 0
4     mult_reference = []
5     candidates = []
6     for tmp in result_list:
7         cand = tmp["response"]
8         ref = tmp["correct_answer"]
9         mult_reference.append([ref])
10        candidates.append(cand)
11        # print("ref", ref)
12        # print("cand", cand)
13
14        rouge_score += scorer.score(ref, cand)['rougeL'].recall
15        cand_split = wordpunc_tokenize(cand)
16        ref_split = wordpunc_tokenize(ref)
17        bleu4_score += sentence_bleu([ref], cand, weights=(0.0, 0.0, 0.0, 1.0))
18        bleu_score += sentence_bleu([ref], cand)
19        meteor_score += meteor_scorer([ref_split], cand_split)
20        # break
21    rouge_score, bleu_score, bleu4_score, meteor_score = rouge_score / len(candidates)
22    P, R, F1 = score(candidates, mult_reference, lang="en", verbose=True)
23    bert_score = R.mean().item()
24    print(f"BLEU Score: {bleu_score}")
25    print(f"BLEU-4 Score: {bleu4_score}")
26    print(f"METEOR Score: {meteor_score}")
27    print(f"ROUGE Score: {rouge_score}")
28    print(f"BERT Score: {bert_score}")
```

- <https://arxiv.org/abs/2402.07729>
 - AIR-Bench: Benchmarking Large Audio-Language Models via Generative Comprehension

[Optional] Alignment on Foundation Benchmark

This is an optional step. This situation applies when your model cannot accurately answer ABCD and needs to be aligned with GPT. We provide a script that can batch call GPT, you only need to do one thing: replace your own GPT call keys (MIT_SPIDER_TOKEN and MIT_SPIDER_URL).

- <https://arxiv.org/abs/2412.13702>
 - Typhoon 2: A Family of Open Text and Multimodal Thai Large Language Models
 - 평가 코드 없음

5.2.3 Experimental Setup

Evaluation: For existing tasks, we use standard metrics.

- <https://github.com/nota-github/audiolm-evaluator>
 - 네이버 커넥트 재단의 부스트캠프 AI Tech 7기 기업 해커톤
 - Nota AI에서 audiolm-evaluator 제작
 - SALMONN의 ASR(WER), AAC(SPIDEr) 평가
- openai/whisper normalizer
- facebookresearch/fairseq tokenizer

```
def compute_spider(hyps, refs) -> float:
    refs = [[ref_] for ref_ in refs] # need to be List[List[str]]
    spider_result = spider(candidates=hyps, mult_references=refs)
    spider_score = round(float(spider_result[0]['spider']),4)
    print(f"SPIDEr: {spider_score}")
```

```
def compute_wer(hyps, refs) -> float:
    normalizer = EnglishTextNormalizer()

    norm_refs = [normalizer(ref) for ref in refs]
    norm_hyps = [normalizer(hyp) for hyp in hyps]

    distance = 0
    ref_length = 0
    tokenizer = EvaluationTokenizer(
        lowercase=True,
        punctuation_removal=True,
        character_tokenization=False,
    )
    for i in range(len(norm_refs)):
        ref = norm_refs[i]
        pred = norm_hyps[i]


        ref_items = tokenizer.tokenize(ref).split()
        pred_items = tokenizer.tokenize(pred).split()

        distance += ed.eval(ref_items, pred_items)
        ref_length += len(ref_items)

    wer = distance / ref_length

    print(f"WER: {wer*100:0.4f}%")
```

- <https://openreview.net/forum?id=14rn7HpKVk>
 - Reference values of the AAC, PR, ER, MC and OSR of the Level 1 tasks are state-of-the-art (SOTA) values.
 - 즉, 그 방식으로 평가하지 않았을 수 있음



Official Comment by Authors

Official Comment by Authors 📅 16 Nov 2023, 01:44 (modified: 20 Nov 2023, 10:59) 👁 Everyone 🔄 Revisions

Comment:

We would like to thank the reviewer for highlighting the value of SALMONN to the community that it unifies the wide range of speech/audio/music tasks and is a useful step towards the research for AGI. We would also like to thank the reviewer for the constructive suggestions which will be responded to one-by-one below. Besides, we also revised parts of the paper (marked in yellow) based on comments from all reviewers.

- Weakness 1: Regarding our choices of reference values, it is only to provide some reasonable references for the readers to understand the tasks and verify the feasibility of our approach.
 - For all the reference values using Whisper, we use the Whisper Large v2 (1550M), whose encoder is also used to construct the SALMONN model.
 - **Reference values of the AAC, PR, ER, MC and OSR of the Level 1 tasks are state-of-the-art (SOTA) values.** Since the encoder of Whisper Large v2 is used in SALMONN, we use its ASR results as the reference value of the ASR task. Indeed, the En2Zh translation result evaluated on the CoVoST2-En2Zh test set might not be the most suitable reference value, since SALMONN was trained on the CoVoST2-En2Zh training set while "Whisper + Vicuna" was not. We thank the reviewer for pointing this out and will update the paper using the BLEU scores provided in [1] where the highest BLEU score of En2Zh is 38.9.

- <https://arxiv.org/abs/2010.14102>
 - Emotion recognition by fusing time synchronous and time asynchronous representations
 - To be consistent and to be able to compare with previous studies, only utterances with ground truth labels belonging to "angry", "happy", "excited", "sad", and "neutral" were used. **The "excited" class was merged with "happy" to better balance** the size of each emotion class, which results in a total of 5,531 utterances (happy 1,636, angry 1,103, sad 1,084, neutral 1,708).
- 논문에서 레퍼런스로 제시한 논문에서 합침

- <https://arxiv.org/abs/2407.09817>
 - Empowering Whisper as a Joint Multi-Talker and Target-Talker Speech Recognition System
 - Permutations with minimum errors are used to compute word error rate (WER) or character error rate (CER) for multitalker ASR as in prior studies [14, 20].
 - [14] <https://arxiv.org/abs/2202.00842>
 - Streaming Multi-Talker ASR with Token-Level Serialized Output Training
 - We evaluated the WER in the same way as the prior work [18] did.
 - [18] <https://arxiv.org/abs/2003.12687>
 - Serialized Output Training for End-to-End Overlapped Speech Recognition
- 우리가 평가한 방식이 적절했음
 - 그러나 salmonn에서 2개의 응답을 내놓지 않으면
 - 응답이 1개 혹은 3개일 경우
 - 정답 2개와 각각 WER 어떻게 비교 및 선택??
 - 건너뛰기??

5.1.2. Evaluation metric

Our trained models were evaluated with respect to WER. In multi-talker ASR, a system may produce a different number of hypotheses than references (i.e., speakers). To cope with this, all possible permutations of the hypothesis order were examined, and the one that yielded the lowest WER was picked.

- prompt
 - Please write down what you hear each person says.
- papers
 - 23.0
- our reproduction
 - 22.1 | 두 개의 문장 응답 비율: 89.43% (2,683 / 3,000)
- 두 개의 문장을 응답한 것에 대해 WER을 측정
 - 이때, 섞인 두 문장에는 순서가 없음
 - $WER((ans1, ans2), (sen1, sen2))$ 와 $WER((ans1, ans2), (sen2, sen1))$ 을 비교해 순서를 정함

- Qwen2-Audio
 - 일본어: ja-mecab
 - 중국어: zh
 - 나머지: 13a

```
if text_lang == "ja":
    text_lang = "ja-mecab"
elif text_lang == "zh":
    text_lang = "zh"
else:
    text_lang = "13a"
```

- AudioBench
 - 중국어: flores101

```
sacrebleu = evaluate.load("sacrebleu")
# Updated to flores101 tokenizer (Thanks Chenyang Lv)
# results = sacrebleu.compute(predictions=predictions, references=references, tokenize='13a')
results = sacrebleu.compute(predictions=predictions, references=references, tokenize='flores101')
```

- 제시된 논문
 - 일본어: char
 - 중국어: char

the last 5 checkpoints for ASR and ST. For MT and ST, we report case-sensitive detokenized BLEU (Papineni et al., 2002) using sacreBLEU (Post, 2018) with default options, except for English-Chinese and English-Japanese where we report character-level BLEU. For ASR, we

- En2Zh: 34.35271 (zh) | Qwen2-Audio
- En2Zh: 35.02756 (char) | cited paper
- En2Zh: 23.06483 (flores101) | AudioBench
- **paper: 33.1**
- En2Zh: 33.47229 (char, normalize, keep eos)
- # skip: 일본어 하나도 없는 경우 (8,929 / 15,531)
- En2Ja: 11.80987 (non-skip, char) | cited paper
- En2Ja: 23.64009 (skip, char) | cited paper
- En2Ja: 6.66087 (non-skip, ja-mecab) | Qwen2-Audio
- En2Ja: 13.97718 (skip, ja-mecab) | Qwen2-Audio
- **paper: 22.7**
- En2Ja: 22.76687 (skip, char, normalize, keep eos)
- SALMONN이 인용한 논문에 따르면
 - For MT and ST, we report case-sensitive detokenized BLEU (Papineni et al., 2002) using sacreBLEU (Post, 2018) with default options, except for **English-Chinese and English-Japanese where we report character-level BLEU**.
 - For all texts, we **normalize the punctuation** and ~...
 - Before calculating WER (CER), sentences are tokenized by sacreBLEU tokenizers, lowercased and with **punctuation removed (except for apostrophes and hyphens)**.

- salmonn 프롬프트는 3개 요청
 - 3개보다 많은 경우에는 주로 inf[:3] (top-k)
- 우리 구현
 - 완전/부분 매칭으로 최대 3개까지 정답으로 인정
 - 정답 / 전체 데이터 * 3
- src: onmt/keyphrase/eval.py

```
if len(pred_list) > cutoff:
    pred_list_k = np.asarray(pred_list[:cutoff])
    match_list_k = match_list[:cutoff]
else:
    pred_list_k = np.asarray(pred_list)
    match_list_k = match_list
```

```
def compute_match_scores(tgt_seqs, pred_seqs, do_lower=True, do_stem=True, type='exact'):
    ...
    If type='exact', returns a list of booleans indicating if a pred has a matching tgt
    If type='partial', returns a 2D matrix, each value v_ij is a float in range of [0,1]
    indicating the (jaccard) similarity between pred_i and tgt_j
```

- WikiQA에 대해 Accuracy로 평가하는 게 드물
 - Microsoft에서 공개한 평가 코드에는 다음의 metric을 평가
 - precision
 - recall
 - fmeasure
- 우리는 LLM-as-Judge로 평가 (논문에 제시된 프롬프트로)
 - Qwen3-8B // ours
 - GPT3.5 // paper
- TTS 모델 미공개 및 평가에 사용된 LLM 차이로 인해 점수가 상이함

| Model | Prec | Rec | F ₁ |
|---------|-------|-------|----------------|
| CNN-Cnt | 26.09 | 37.04 | 30.61 |
| +QLen | 27.96 | 37.86 | 32.17 |
| +SLen | 26.14 | 37.86 | 30.92 |
| +QClass | 27.84 | 33.33 | 30.34 |
| +All | 28.34 | 35.80 | 31.64 |

Table 5: Evaluation of answer triggering on the WIKIQA dataset. Question-level precision, recall and F₁ scores are reported.

Table 2: Test sets, metrics, and sources of the reference values used in the three levels of tasks.
 The speech data used in SQQA and KE are synthesised using a commercial text-to-speech product.

To evaluate answers of the model of spoken-query-based question answering (SQQA).

Next I will give you a question and give you the corresponding standard answer and the answer I said. You need to judge whether my answer is correct or not based on the standard answer to the question. I will give you the question and the corresponding answer in the following form: {'Question': 'xxx', 'Standard Answer': 'xxx', 'My Answer': 'xxx'} \n You need to judge the correctness of my answer, as well as state a short justification. Your responses need to follow the Python dictionary format: \n {"Correct": True / False, "Reason": "xxx"} \n Now, I will give you the following question and answer: SENTENCEHERE \n Your response is:

- 우리는 LLM-as-Judge로 평가
 - prompt: "According to the speech, what is the {}?"

| answer | inferred |
|-----------|--|
| meeting | the event_name is 'sesame meeting next tuesday at eleven a.m. with jessie'. |
| tuesday | the date is next tuesday at 11 a.m. |
| eleven am | the time is 11 am on tuesday. |
| jesse | the person is not specified in the given sentence. |

```
1 {
2   "idx": 0,
3   "id": 9054,
4   "path": "/home/jpong/Workspace/jaeewon/slurp/wav/00000_09054.wav",
5   "slots": "event_name=mona; date=tuesday",
6   "transcript": "event reminder mona tuesday"
7 }
```

```
USER_PROMPT = """
determine if the 'our_slot' accurately reflects the 'golden_slot' and respond only with 'Yes' or 'No'.
you can refer to the 'slot_type' for better understanding.

- **our_slot**: `{our_slot}`
- **golden_slot**: `{golden_slot}`
- **slot_type**: `{slot_type}`
"""

SYSTEM_PROMPT = """
you will be given two slots: 'our_slot' and 'golden_slot', along with a 'slot_type' that describes the nature of the slot.
you don't need to flatter for our model and you don't need to be harsh for our model.
you must respond 'Yes' if 'our_slot' accurately reflects 'golden_slot', otherwise respond 'No'.
you must respond strictly with either 'Yes' or 'No'.
you must not include any explanations or additional text.
"""
```

- <https://arxiv.org/abs/2412.13702>
 - Typhoon 2: A Family of Open Text and Multimodal Thai Large Language Models
 - We observed examples where Typhoon2-Audio **did not provide any answer** to speech instruction in nested commands; **hence, receiving very low scores on these examples.**
 - 여기서는 이상치를 모두 포함한 것으로 보임
- SALMONN의 WER 성능을 자기네들 방식으로 측정하여 제시

5.2.3 Experimental Setup

Evaluation: For existing tasks, we use standard metrics.

| Model | Size | ASR (WER↓) | | Translation (BLEU↑) | | | Gender (Acc↑) | |
|----------------|------|-------------|--------------|---------------------|--------------|--------------|---------------|--------------|
| | | En | Th | Th2En | En2Th | X2Th | En | Th |
| Qwen-Audio | 7B | 6.94 | 95.12 | 0.00 | 2.48 | 0.29 | 37.09 | 67.97 |
| SALMONN | 13B | 5.79 | 98.07 | 14.97 | 0.07 | 0.10 | 95.69 | 93.26 |
| DiVA | 8B | 30.28 | 65.21 | 7.97 | 9.82 | 5.31 | 47.30 | 50.12 |
| Gemini-1.5-Pro | - | 5.98 | 13.56 | 22.54 | 20.69 | 13.52 | 90.73 | 81.32 |
| Typhoon-Audio | 8B | 8.72 | 14.17 | 24.14 | 17.52 | 10.67 | 98.76 | 93.74 |
| Typhoon2-Audio | 8B | 5.83 | 14.04 | 33.25 | 27.15 | 15.93 | 76.51 | 75.65 |

Table 31: Audio LM Evaluation in English and Thai on ASR, Translation, Gender Classification. Size refers to the size of the LLM.

- 논문은 데이터셋, 지표 그리고 수치만 제시함
- 각 태스크에 대해, 코드를 공개한 논문들 간에도 평가 방법론이 상이함
- GPT로 alignment하고 평가하는 경우도 있음
- 자기네들 방식으로 기존 모델을 평가했으나, 코드를 공개하지 않는 경우도 있음
- TT

reproduce
SALMONN &
SAKURA

학생 원종찬, 최재원
Language & AI융합전공

2025. 10. 27. 3 PM.
한국외대 교수회관 401호

- <https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/>
 - batch-invariant
 - batch가 같다면 그것들 간에는 deterministic함을 확인함
 - salmonn에서 추론시 batch가 1이라고 명시적으로 언급한 적 없음

```
==== batch=1 ====
[' Ducks quacking.</s>']
elapsed 1.5202248480636626s

==== batch=2 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 1.1303708120249212s

==== batch=3 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 1.373612365918234s

==== batch=4 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 1.06810438890788s

==== batch=5 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 1.3848115589935333s

==== batch=6 ====
[' Ducks quacking.</s>', ' Ducks quacking.</s>', ' Ducks quacking.</s>', ' Ducks quacking.</s>', ' Ducks quacking.</s>', ' Ducks quacking.</s>']
elapsed 1.5191025659441948s

==== batch=7 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 1.7930845129303634s

==== batch=8 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 1.7784668831154704s

==== batch=9 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 1.873016953933984s

==== batch=10 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 2.1175767299719155s

==== batch=11 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 2.047229261137545s

==== batch=12 ====
[' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>', ' You hear a duck call.</s>']
elapsed 2.076617281883955s
```

Hankuk University of Foreign Studies

- <https://arxiv.org/abs/2406.16020v4>

- AudioBench: A Universal Benchmark for Audio Large Language Models

- For the CommonVoice dataset, SALMONN tends to perform speech translation on a significant number of samples, which adversely affects its performance on the Word Error Rate (WER) metric. This suggests that SALMONN may be overly tuned to speech features (tokens) and not sufficiently responsive to the prompts.

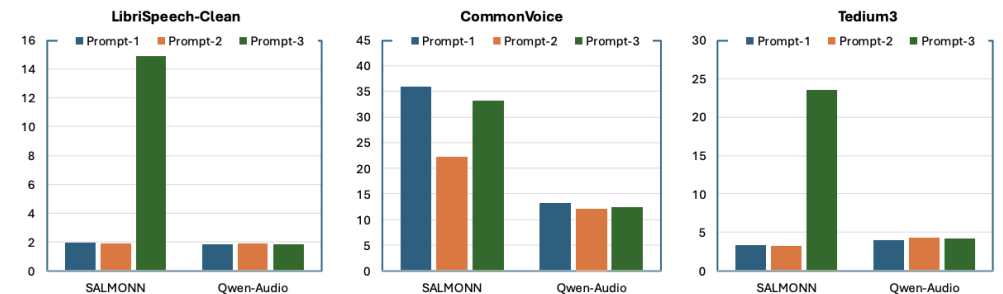


Figure 2: Sensitivity tests were conducted on three different prompts on the same datasets. Performance is measured by word error rate (WER)_(↓). The results state that SALMONN is the least robust when faced with varying prompt templates for the same task. Prompt 1-3: {"Turn the speech input into a text transcription", "Recognize the speech and give me the transcription", "Decode the audio and give me the written transcriptions."}