AUDIO DIFFERENCE CAPTIONING UTILIZING SIMILARITY-DISCREPANCY DISENTANGLEMENT

Daiki Takeuchi, Yasunori Ohishi, Daisuke Niizumi, Noboru Harada, and Kunio Kashino

NTT Corporation, Japan

ABSTRACT

We proposed *Audio Difference Captioning* (ADC) as a new extension task of audio captioning for describing the semantic differences between input pairs of similar but slightly different audio clips. The ADC solves the problem that conventional audio captioning sometimes generates similar captions for similar audio clips, failing to describe the difference in content. We also propose a cross-attention-concentrated transformer encoder to extract differences by comparing a pair of audio clips and a similarity-discrepancy disentanglement to emphasize the difference in the latent space. To evaluate the proposed methods, we built an AudioDiffCaps dataset consisting of pairs of similar but slightly different audio clips with human-annotated descriptions of their differences. The experiment with the AudioDiffCaps dataset showed that the proposed methods solve the ADC task effectively and improve the attention weights to extract the difference by visualizing them in the transformer encoder.

Index Terms: audio difference captioning, contrastive learning, crossmodal representation learning, deep neural network

1. INTRODUCTION

Audio captioning is used to generate the caption for an audio clip [1–10]. Unlike labels for scenes and events [11–15], captions describe the content of the audio clip in detail. However, conventional audio captioning systems often produce similar captions for similar audio clips, making it challenging to discern their differences solely based on the generated captions. For instance, suppose two audio clips of heavy rain are input into a conventional captioning system. The system will generate a caption describing the content of each, like "*It is raining very hard without any break*" and "*Rain falls at a constant and heavy rate*"[1] as illustrated in Fig. 1(a). The difference, such as which rain sound is louder, is difficult to understand from the generated captions in this case.

To address this problem, we propose *Audio Difference Captioning* (ADC) as a new extension task of audio captioning. ADC takes two audio clips as input and outputs text explaining the difference between two inputs as shown in Fig. 1. We make the ADC clearly describe the difference between the two audio clips, such as "Make the rain louder," which describes what and how to modify one audio clip to the other in the instruction form, even for audio clips with similar texts. Potential real-world applications include machine condition and healthcare monitoring using sound by captioning anomalies that differ from usual sounds.

The ADC task has two major challenges: different content detection and detection sensitivity. Since the difference between a pair of audio clips can be classes of contained events or an attribute, such as loudness, the ADC needs to detect what difference to describe. When the difference lies in an attribute, the ADC needs to be sensitive enough to detect the magnitude of the attribute, such as rain is hard or moderately shown in the example in Fig. 1.

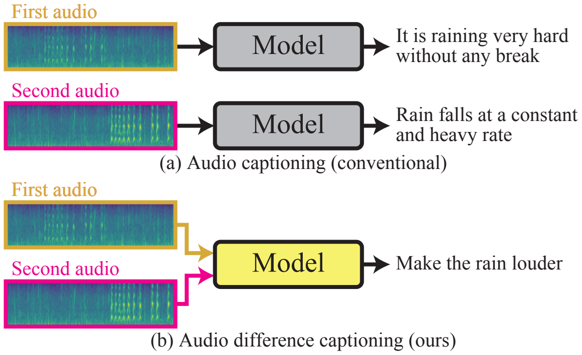

First audio
Model
It is raining very hard without any break
Second audio
Model
Rain falls at a constant and heavy rate
(a) Audio captioning (conventional)
First audio
Second audio
Model
Make the rain louder
(b) Audio difference captioning (ours)

Figure 1: Conceptual diagram of conventional audio captioning and audio difference captioning. Audio difference captioning describes the difference between pair audio clips, while conventional audio captioning describes the contents of each.

To handle these challenges, the ADC should extract features of difference based on the cross-reference of two audio clips. These features should carry enough information to differentiate critical attributes such as loudness. A typical choice of a feature extractor could be pre-trained models to classify labels [16–18]. However, these models learn to discriminate sound event classes, learning what is common while ignoring subtle differences such as raining hard or quietly unless the class definition covers that.

To meet the requirements of the ADC mentioned above, we propose (I) a cross-attention-concentrated (CAC) transformer encoder and (II) a similarity-discrepancy disentanglement (SDD). The CAC transformer encoder utilizes the masked multi-head attention layer, which only considers the cross-attention of two audio clips to extract features of difference efficiently. The SDD emphasizes the difference feature in the latent space using contrastive learning based on the assumption that two similar audio clips consist of similar and discrepant parts.

We demonstrate the effectiveness of our proposals using a newly built dataset, AudioDiffCaps, consisting of two similar but slightly different audio clips synthesized from existing environmental sound datasets [11, 15] and human-annotated difference descriptions. Experiments show that the CAC transformer encoder improves the evaluation metric scores by making the attention focus only on cross-references. The SDD also improves the scores by emphasizing the differences between audio clips in the latent space. Our contributions are proposals of (i) the ADC task, (ii) the CAC transformer encoder and SDD for solving ADC, (iii) the AudioD-

[1] These captions were taken from the Clotho dataset [2]

iffCaps dataset, and (iv) demonstrating the effectiveness of these proposals.

2. AUDIO DIFFERENCE CAPTIONING

We propose ADC, a task for generating texts to describe the difference between two audio clips. ADC estimates a word sequence $\boldsymbol{w}$ from the two audio clips $\boldsymbol{x}$ and $\boldsymbol{y}$.

The general framework to solve ADC includes three main functions: audio embedding, audio difference encoding, and text decoding. Audio embedding calculates two audio embedding vectors from two audio clips, respectively. Audio difference encoding captures the difference between two audio embedding vectors. Text decoding generates a description of the differences from captured differences. Audio embedding and audio difference encoding require approaches specific to ADC. In particular, difference encoding is the function unique to audio difference captioning. This function requires a model structure to capture the subtle differences between two audio clips, unlike conventional audio captioning that captures the content of a single audio clip. Moreover, the sensitivity to the subtle difference between two similar audio clips is also necessary for audio embedding. The pre-trained audio embedding models widely used for conventional environmental sound analysis tasks are often trained for classification tasks and are suitable for identifying predefined labels. Consequently, the outputs of these pre-trained audio embedding models are not sensitive to the subtle differences between audio clips with the same label. Therefore, learning to emphasize the differences between similar audio clips in the latent space is necessary when applying pre-trained audio embedding models to the ADC.

3. PROPOSED METHOD

Based on the above discussion, we propose the ADC system illustrated in Fig. 2. Our system consists of an audio feature extractor (red), difference encoder (blue), text decoder (green), and similarity-discrepancy disentanglement (purple).

3.1. Audio feature extractor

The audio feature extractor uses a pre-trained audio embedding model to calculate audio embedding vectors. Two audio clips $\boldsymbol{x}$ and $\boldsymbol{y}$ are the input, and the audio embedding vectors corresponding to the clips $\boldsymbol{X} \in \mathbb{R}^{H \times T_x}$ and $\boldsymbol{Y} \in \mathbb{R}^{H \times T_y}$ are the output, where H is the size of hidden dimension, T_x is the time length of $\boldsymbol{X}$, and T_y is the time length of $\boldsymbol{Y}$

3.2. Difference encoder

The difference encoder extracts information about the differences between the two audio clips from audio embedding vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$. To extract difference information efficiently, we utilize a cross-attention-concentrated (CAC) transformer encoder as the main function of the difference encoder. The CAC transformer encoder utilizes the masked multi-head attention layer, allowing only mutual cross-attention between two audio clips by the attention mask illustrated in the upper right of Fig. 2.

The detailed procedure is as follows. First, special tokens that indicate the order of the audio clips $\mathcal{X} \in \mathbb{R}^{H \times 1}$ and $\mathcal{Y} \in \mathbb{R}^{H \times 1}$ are concatenated at the beginning of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively. Next, these two sequences are concatenated to make the input of the difference encoder $\boldsymbol{Z}$ like $\boldsymbol{Z} = [\mathcal{X}, \boldsymbol{X}, \mathcal{Y}, \boldsymbol{Y}]$. Then, positional encoding $\mathscr{P}$ is applied to $\boldsymbol{Z}$. Finally, $\mathscr{P}(\boldsymbol{Z})$ is input to CAC transformer encoder to obtain the output $\hat{\boldsymbol{Z}} = [\hat{\mathcal{X}}, \hat{\boldsymbol{X}}, \hat{\mathcal{Y}}, \hat{\boldsymbol{Y}}]$.

3.3. Text decoder

The transformer decoder is utilized as a text decoder like as [5]. The text decoder calculates word probability from the output of the difference encoder $\hat{\boldsymbol{Z}}$.

3.4. Similarity-discrepancy disentanglement

The similarity-discrepancy disentanglement (SDD) loss function is an auxiliary loss function aimed at obtaining a difference-emphasized audio representation. When there is an explainable difference between two audio clips, these clips consist of similar and discrepant parts. To introduce this hypothesis, we design contrastive learning to bring similar parts closer and keep discrepant parts. We propose two types of implementations that apply SDD to the input of the difference encoder $\boldsymbol{Z}$ or the output of it $\hat{\boldsymbol{Z}}$, as shown in Fig. 2, and call the former and latter implementations early and late disentanglement, respectively.

We explain the procedure in the case of early disentanglement. Note that the case of late disentanglement only replaces $\boldsymbol{Z}$ with $\hat{\boldsymbol{Z}}$. First, $\boldsymbol{Z}$ is split along the hidden dimension and assigned to similar and discrepant parts like in the upper left illustration of Fig. 2. If $\boldsymbol{Z} \in \mathbb{R}^{H \times (T_x + T_y + 2)}$, $\boldsymbol{Z}$ is split into similar part $\boldsymbol{Z}_{\mathrm{S}}$ and discrepant part $\boldsymbol{Z}_{\mathrm{D}}$ like

$$\boldsymbol{Z}_{\mathrm{S}} = [\mathcal{X}_{\mathrm{S}}, \boldsymbol{X}_{\mathrm{S}}, \mathcal{Y}_{\mathrm{S}}, \boldsymbol{Y}_{\mathrm{S}}] \in \mathbb{R}^{(H/2) \times (T_x + T_y + 2)}, \tag{1}$$

$$\boldsymbol{Z}_{\mathrm{D}} = [\mathcal{X}_{\mathrm{D}}, \boldsymbol{X}_{\mathrm{D}}, \mathcal{Y}_{\mathrm{D}}, \boldsymbol{Y}_{\mathrm{D}}] \in \mathbb{R}^{(H/2) \times (T_x + T_y + 2)}. \tag{2}$$

Then, the SDD is performed by $\mathcal{L}_{\mathrm{SDD}} = \mathcal{L}_{\mathrm{S}} + \mathcal{L}_{\mathrm{D}}$, where

$$\mathcal{L}_{\mathrm{S}} = \mathrm{SymInfoNCE}(\Phi([\mathcal{X}_{\mathrm{S}}, \boldsymbol{X}_{\mathrm{S}}]), \Phi([\mathcal{Y}_{\mathrm{S}}, \boldsymbol{Y}_{\mathrm{S}}])), \tag{3}$$

$$\mathcal{L}_{\mathrm{D}} = \mathrm{PairCosSim}(\Psi([\mathcal{X}_{\mathrm{D}}, \boldsymbol{X}_{\mathrm{D}}]), \Psi([\mathcal{Y}_{\mathrm{D}}, \boldsymbol{Y}_{\mathrm{D}}])), \tag{4}$$

SymInfoNCE is the symmetric version of the InfoNCE loss used in [19], PairCosSim is the cosine similarity for each correct data pair, Φ and Ψ are embedding networks consisting of the bidirectional-LSTM and average pooling, and $\mathcal{L}_{\mathrm{SDD}}$ is the final value of the SDD loss function. That is, the SDD loss function views $[\mathcal{X}_{\mathrm{S}}, \boldsymbol{X}_{\mathrm{S}}]$ and $[\mathcal{Y}_{\mathrm{S}}, \boldsymbol{Y}_{\mathrm{S}}]$ as similar parts and brings them closer by using $\mathcal{L}_{\mathrm{S}}$ and views $[\mathcal{X}_{\mathrm{D}}, \boldsymbol{X}_{\mathrm{D}}]$ and $[\mathcal{Y}_{\mathrm{D}}, \boldsymbol{Y}_{\mathrm{D}}]$ as discrepant parts and keeps them apart by $\mathcal{L}_{\mathrm{D}}$.

The entire loss function $\mathcal{L}$ is the weighted sum of cross-entropy loss for word prediction $\mathcal{L}_{\mathrm{CE}}$ and the SDD: $\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \lambda \mathcal{L}_{\mathrm{SDD}}$, where λ is a weighting parameter.

4. EXPERIMENT

Experiments were conducted to evaluate the proposed CAC transformer encoder and SDD loss function. We constructed the AudioDiffCaps dataset consisting of pairs of similar but slightly different audio clips and a human-annotated description of their differences for the experiments.

4.1. AudioDiffCaps dataset

The constructed AudioDiffCaps dataset consists of (i) pairs of similar but slightly different audio clips and (ii) human-annotated descriptions of their differences.

The pairs of audio clips were artificially synthesized by mixing foreground event sounds with background sounds taken from existing environmental sound datasets (FSD50K [15] and ESC-50 [11])

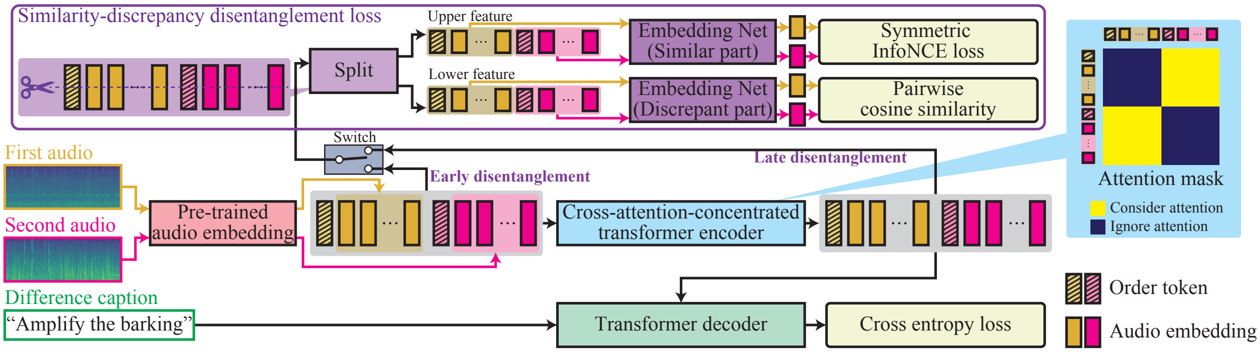


Figure 2: Model architecture of our proposed method. The cross-attention-concentrated transformer encoder uses an attention mask illustrated in the upper left. The similarity-discrepancy disentanglement is conducted by symmetric InfoNCE loss and pairwise cosine similarity. The input to them is either the input or output of the cross-attention-concentrated transformer encoder.

using the Scaper library for soundscape synthesis and augmentation [20]. We used the same mixing procedure as our previous work [21]. Data labeled *rain* or *car_passing_by* in FSD50K was used as background, and six foreground event classes were taken from ESC-50 (i.e., data labeled *dog*, *chirping_bird*, *thunder*, *footsteps*, *car_horn*, and *church_bells*). Each created audio clip was 10 seconds long. The maximum number of events in one audio clip was two, with 0-100% overlap (no overlap-range control applied). Each foreground event class had 32 or 8 instances in the development or evaluation set, respectively. Similar to previous work, we focused on the three types of difference: increase/decrease of background sounds, increase/decrease of sound events, and addition/removal of sound events. The development and evaluation sets contained 5996 and 1720 audio clip pairs, respectively. (That is, development and evaluation sets contained 11992 and 3440 audio clips.)

The human-annotated descriptions were written as instruction forms explaining "what and how" to change the first audio clip to create the second audio clip. In the preliminary study, we found that declarative sentences, in some cases, tend to use ordinal numbers such as "*First sound* is louder than *second sound*". Since these cases do not express what the actual difference is, the AudioDiffCaps dataset uses instruction forms with a fixed direction of change from the first audio clip to the second one, e.g., "Make the rain louder" [2] . A wider variety of descriptions explaining the same concept, such as declarative sentences, should be included in future works. The presentation order of the pair to the annotator was randomly selected. Annotators were five naïve workers remotely supervised by an experienced annotator. Each pair of audio clips in the development set had between 1 and 5 descriptions (a total of 28,892) while each pair in the evaluation set had exactly five descriptions assigned to it (a total of 8600).

4.2. Experimental conditions

We used 10% of the development set for validation. The optimizer was Adam [22]. The number of epochs was 100. We used the BLEU-1, BLEU-4, METEOR, ROUGE-L, CIDEr [23], SPICE [24], and SPIDEr [25] as evaluation metrics. They were also used for conventional audio captioning [26].

We used BYOL-A [27], a pre-trained audio embedding model, as the audio feature extractor in our ADC implementation, and we fine-tuned the BYOL-A throughout experiments. The transformer encoder and decoder used the official implementation of PyTorch. The number of layers was 1. The hidden size was 768. The number of heads was 4. The activation was RELU. The dimension of the feedforward layer was 512. The dropout rate was 0.1. For the attention mask of the transformer encoder, we compared two types; one with the proposed cross-attention mask and the other without a mask. The text decoder used the teacher forcing algorithm during training and the beam search algorithm [28, 29] during inference. The value of λ was empirically set to 0, 0.5, 1.0, or 2.0.

4.3. Results

The results of evaluation metrics are shown in Table 1, where bold font indicates the highest score, "Mask" and "Disent." indicate the attention mask utilized in the transformer encoder and input of SDD loss function, respectively. When the CAC transformer encoder was evaluated by comparing the two lines above, the proposed method had superior or equivalent scores to the conventional method in all evaluation metrics. There was no significant difference in the evaluation metrics related to the degree of matching with single-word references, such as BLEU-1. One likely reason is that the scores above a certain level can be obtained by outputting words in arbitrary sentences, such as "a" and "the" in these metrics. In contrast, the scores of BLEU-4, ROUGE-L, CIDEr, and SPIDEr, affected by the accuracy of consecutive words, were improved using the proposed cross-attention mask. Therefore, the proposed cross-attention mask was thought to make the feature extraction of differences more efficient and simplify the training of the text decoder. As a result, phrase-level accuracy was improved.

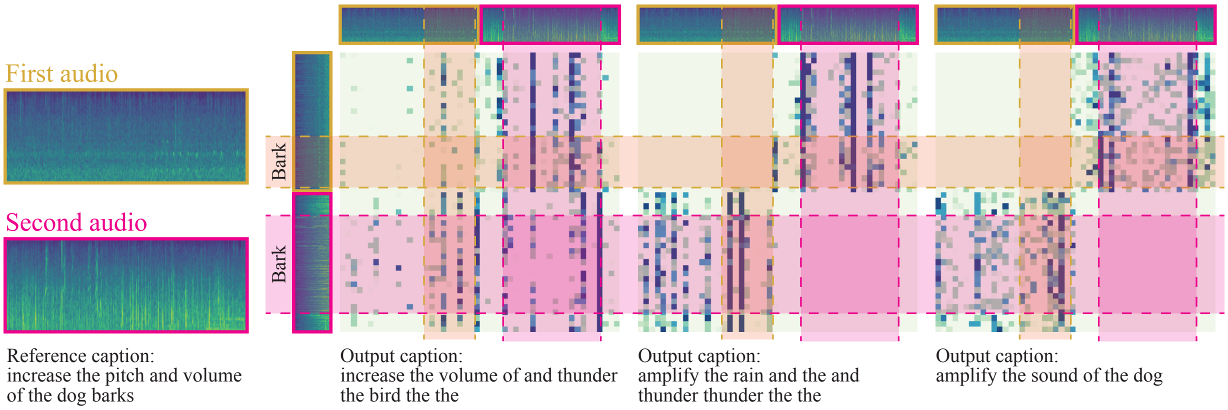
The effect of SDD was verified from the results of the second to eighth lines. The results in (a) and (b) were the conventional transformer without cross attention mask or SDD loss and the CAC transformer without SDD loss ($\lambda = 0$) Ones from (c) to (h) were the result when using early/late disentanglement. Since the scores of BLEU-4, ROUGE-L, CIDEr, and SPIDEr improved under all conditions comparing (b) and others, the SDD loss function was effective for the audio difference captioning task. The improvement in the case of late disentanglement (f), (g), and (h) was remarkable, and the results obtained the best scores in all evaluation metrics with late disentanglement. In other words, it was essential to use the information to be compared to decompose the similar part and the different parts in the feature amount space. That corresponds to the difference determined depending on the comparison target.

Fig. 3 shows one of the evaluation data and estimated caption and attention weight of the transformer encoder from each system. The leftmost colomn is the Mel-spectrogram of the two input audio

[2] The dataset is available at `https://github.com/nttcslab/audio-diff-caps`.

Table 1: Results of evaluation metrics

ID	System	Mask	Disent.	BLEU-1	BLEU-4	METEOR	ROUGE-L	CIDEr	SPICE	SPIDEr
(a)	Baseline	N/A	N/A	67.1	31.7	24.3	56.9	82.7	19.5	51.1
(b)	CAC transformer	Cross	N/A	67.0	33.4	25.2	59.5	90.2	19.5	54.9
	CAC transformer									
(c)	w/ Early SDD ($\lambda = 0.5$)	Cross	Early	67.0	33.7	25.3	59.6	91.8	19.4	55.6
(d)	w/ Early SDD ($\lambda = 1.0$)	Cross	Early	66.8	32.2	25.3	59.3	91.7	19.5	55.6
(e)	w/ Early SDD ($\lambda = 2.0$)	Cross	Early	66.9	33.5	25.3	59.6	92.8	18.7	55.8
(f)	w/ Late SDD ($\lambda = 0.5$)	Cross	Late	**70.3**	39.2	**26.4**	**61.6**	**97.6**	21.3	59.4
(g)	w/ Late SDD ($\lambda = 1.0$)	Cross	Late	69.9	38.3	26.3	61.5	96.3	21.2	58.7
(h)	w/ Late SDD ($\lambda = 2.0$)	Cross	Late	69.9	**39.5**	26.3	61.3	97.1	**22.6**	**59.9**



(a) Baseline

(b) Cross-attention-concentrated transformer

(h) Cross-attention-concentrated transformer w/ Late disentanglement(λ=2.0)

Figure 3: Examples of output caption and attention weights. The leftmost row was the Mel-spectrogram of two audio clips and one reference caption. The three on the right were the attention weights of the transformer encoder and the output caption.

clips and one of the reference captions. The three on the right are the attention weight of the transformer encoder and output caption, where the attention weight shows the average of multiple heads. The audio clips on the left and above the weights correspond to the input and memory of the transformer, respectively. The area colored pink and yellow on the weights corresponds to the dog barking. Since there was a difference in the loudness of the dog barking between the two clips, the attention was expected to focus on areas where pink and yellow overlap to extract the difference.

First, in (a), since the attention weight was not constrained, it was also distributed widely to areas other than the above compared with the other two. On the other hand, the attention weights of (b) and (h) concentrated on areas where pink and yellow overlap since the attention of the same input and memory was unavailable. Comparing (b) and (h), while the attention of the part containing the barking of the dog in the memory was large at any time-frame in (b), more attention was paid to the pink and yellow overlapping areas where both input and the memory contain the barking of the dog in (h). Since the late disentanglement required that similar and discrepant parts be retained in the output of the transformer encoder calculated using these attention weights, it was thought that the late disentanglement induced attention to be paid to the part where there was a difference when comparing the two sounds instead of paying attention to the parts that are likely to exist the difference compared with the distribution of training data, such as a dog barking.

5. CONCLUSION

We proposed *Audio Difference Captioning* (ADC) as a new extension task of audio captioning for describing the semantic differences between similar but slightly different audio clips. The ADC solves the problem that conventional audio captioning sometimes generates similar captions for similar but slightly different audio clips, failing to describe the difference in content. We also propose a cross-attention-concentrated transformer encoder to extract differences by comparing a pair of audio clips and a similarity-discrepancy disentanglement to emphasize the difference feature in the latent space. To evaluate the proposed methods, we newly built an AudioDiffCaps dataset consisting of pairs of similar but slightly different audio clips and a human-annotated description of their differences. We experimentally showed that since the attention weights of the cross-attention-concentrated transformer encoder are restricted only to the mutual direction of the two inputs, the differences can be efficiently extracted. Thus, the proposed method solved the ADC task effectively and improved the evaluation metric scores.

Future work includes utilizing a pre-trained generative language model such as BART [30] and applying a wider variety of audio events and types of differences.

6. ACKNOWLEDGMENTS

BAOBAB Inc. supported the annotation for the dataset.

7. REFERENCES

[1] C. D. Kim, B. Kim, H. Lee, and G. Kim, "AudioCaps: Generating captions for audios in the wild," in *Proc. Conf. N. Am. Chapter Assoc. Comput. Linguist.*, 2019, pp. 119–132.

[2] K. Drossos, S. Adavanne, and T. Virtanen, "Clotho: An audio captioning dataset," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2019, pp. 736–740.

[3] D. Takeuchi, Y. Koizumi, Y. Ohishi, N. Harada, and K. Kashino, "Effects of word-frequency based pre- and post-processings for audio captioning," in *Proc. Detect. Classif. Acoust, Scenes Events Workshop (DCASE)*, November 2020.

[4] X. Xu, H. Dinkel, M. Wu, and K. Yu, "A crnn-gru based reinforcement learning approach to audio captioning." in *Proc. Detect. Classif. Acoust. Scenes Events (DCASE) Workshop*, 2020, pp. 225–229.

[5] X. Mei, X. Liu, Q. Huang, M. D. Plumbley, and W. Wang, "Audio captioning transformer," in *Proc. Detect. Classif. Acoust. Scenes Events (DCASE) Workshop*.

[6] F. Gontier, R. Serizel, and C. Cerisara, "Automated audio captioning by fine-tuning bart with audioset tags," in *Proc. Detect. Classif. Acoust. Scenes Events (DCASE) Workshop*, 2021.

[7] Y. Koizumi, Y. Ohishi, D. Niizumi, D. Takeuchi, and M. Yasuda, "Audio captioning using pre-trained large-scale language model guided by audio-based similar caption retrieval," *arXiv preprint arXiv:2012.07331*, 2020.

[8] X. Xu, M. Wu, and K. Yu, "Diversity-controllable and accurate audio captioning based on neural condition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2022, pp. 971–975.

[9] X. Mei, X. Liu, J. Sun, M. D. Plumbley, and W. Wang, "Diverse audio captioning via adversarial training," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2022, pp. 8882–8886.

[10] X. Liu, X. Mei, Q. Huang, J. Sun, J. Zhao, H. Liu, M. D. Plumbley, V. Kilic, and W. Wang, "Leveraging pre-trained bert for audio captioning," in *Proc. 24th Eur. Signal Process. Conf. (EUSIPCO)*, 2022, pp. 1145–1149.

[11] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proc. 23rd Annual ACM Conf. Multimedia*, pp. 1015–1018.

[12] D. Barchiesi, D. Giannoulis, D. Stowell, and M. D. Plumbley, "Acoustic scene classification: Classifying environments from the sounds they produce," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 16–34, 2015.

[13] A. Mesaros, T. Heittola, and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," in *Proc. 24th Eur. Signal Process. Conf. (EUSIPCO)*. IEEE, 2016, pp. 1128–1132.

[14] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*. IEEE, 2017, pp. 776–780.

[15] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "Fsd50k: an open dataset of human-labeled sound events," *arXiv preprint arXiv:2010.00475*, 2020.

[16] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*. IEEE, 2017, pp. 131–135.

[17] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 28, pp. 2880–2894, 2020.

[18] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio spectrogram transformer," in *Proc. Interspeech*, 2021, pp. 571–575.

[19] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," *arXiv preprint arXiv:2103.00020*, 2021.

[20] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust. (WASPAA)*. IEEE, 2017, pp. 344–348.

[21] D. Takeuchi, Y. Ohishi, D. Niizumi, N. Harada, and K. Kashino, "Introducing auxiliary text query-modifier to content-based audio retrieval," in *Proc. Interspeech*, 2022.

[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. in Int. Conf. Learn. Represent. (ICLR)*, 2014.

[23] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, "CIDEr: Consensus-based image description evaluation," in *IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 4566–4575.

[24] P. Anderson, B. Fernando, M. Johnson, and S. Gould, "SPICE: Semantic propositional image caption evaluation," in *Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 382–398.

[25] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, "Improved image captioning via policy gradient optimization of spider," in *IEEE Int. Conf. Comput.Vis. (ICCV)*, 2017, pp. 873–881.

[26] DCASE2022 Challenge Task 6: Automated Audio Captioning and Language-Based Audio Retrieval, https://dcase.community/challenge2022/task-automatic-audio-captioning-and-language-based-audio-retrieval.

[27] D. Niizimi, D. Takeuchi, Y. Ohishi., N. Harada, and K. Kashino, "BYOL for audio: Self-supervised learning for general-purpose audio representation," in *Proc. Int. Jt. Conf. Neural Netw. (IJCNN)*. IEEE, 2021.

[28] P. Koehn, *Statistical machine translation*. Cambridge University Press, 2009.

[29] ——, "Pharaoh: a beam search decoder for phrase-based statistical machine translation," in *Mach. Transl.: Real Users Res. Proc. 6th Conf. Assoc. Mach. Transl. Am. (AMTA-2004)*, vol. 3265, 2004.

[30] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proc. 58th Annu. Meet. Assoc. Comput. Linguist.*, 2020, pp. 7871–7880.