INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

A Transformer-based Audio Captioning Model with Keyword Estimation

Yuma Koizumi, Ryo Masumura, Kyosuke Nishida, Masahiro Yasuda, and Shoichiro Saito

NTT Corporation, Japan

koizumi.yuma@ieee.org

Abstract

One of the problems with automated audio captioning (AAC) is the indeterminacy in word selection corresponding to the audio event/scene. Since one acoustic event/scene can be described with several words, it results in a combinatorial explosion of possible captions and difficulty in training. To solve this problem, we propose a Transformer-based audio-captioning model with keyword estimation called *TRACKE*. It simultaneously solves the word-selection indeterminacy problem with the main task of AAC while executing the sub-task of acoustic event detection/acoustic scene classification (i.e., keyword estimation). TRACKE estimates keywords, which comprise a word set corresponding to audio events/scenes in the input audio, and generates the caption while referring to the estimated keywords to reduce word-selection indeterminacy. Experimental results on a public AAC dataset indicate that TRACKE achieved state-of-the-art performance and successfully estimated both the caption and its keywords.

Index Terms: automated audio captioning, keyword estimation, audio event detection, and Transformer.

1. Introduction

Automated audio captioning (AAC) is an intermodal translation task when translating an input audio into its description using natural language [1–6]. In contrast to automatic speech recognition (ASR), which converts a speech to a text, AAC converts environmental sounds to a text. This task potentially raises the level of automatic understanding of sound environment from merely tagging events [7, 8] (e.g. alarm), scenes [9] (e.g. kitchen) and condition [10] (e.g. normal/anomaly) to higher contextual information including concepts, physical properties, and high-level knowledge. For example, a smart speaker with an AAC system will be able to output "*a digital alarm in the kitchen has gone off three times*," and might give us more intelligent recommendations such as "*turn the gas range off.*"

One of the problems with AAC is the existence of many possible captions that correspond to an input. In ASR, a set of phonemes in a speech corresponds almost one-to-one to a word. In contrast, one acoustic event/scene can be described with several words, such as {*car, automobile, vehicle, wheels*} and {*road, roadway, intersection, street*}. Such indeterminacy in word selection leads to a combinatorial explosion of possible answers, making it almost impossible to estimate the ground-truth and difficulty in training an AAC system.

To reduce the indeterminacy in word selection, conventional AAC setups allow the use of keywords related to acoustic events/scenes [4, 5]. The audio samples in the AudioCaps dataset [4] are parts of the Audio Set [11], and their captions are annotated while referring to the Audio Set labels. Therefore, automatic text generation while referring to keywords (e.g. Audio Set label) may restrict the solution space and should be effective in reducing word-selection indeterminacy.

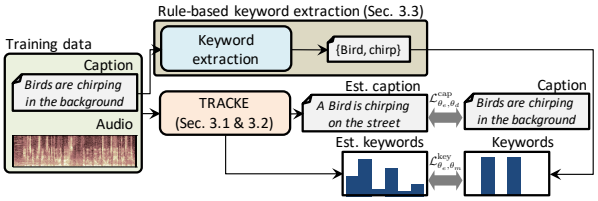

Figure 1: *Overview of training procedure of TRACKE.*

Unfortunately, in some real-world applications such as using a smart speaker, it is difficult to provide such keywords in advance. For example, to output the caption "*a digital alarm in the kitchen has gone off three times*," conventional AAC systems require the keywords related to the acoustic events/scenes such as {alarm, kitchen}. However, if the user can input such keywords, the user should know the sound environment without any captions. This dilemma means that we need to solve the word-selection indeterminacy problem of AAC while simultaneously executing the traditional sub-task of acoustic event detection (AED) [7, 8]/acoustic scene classification (ASC) [9][1].

We propose a Transformer [13]–based audio captioning model with keyword estimation called *TRACKE*, which simultaneously solves the word-selection indeterminacy problem of AAC and executing the AED/ASC sub-task (i.e. keyword estimation). Figure 1 shows an overview of the training procedure of TRACKE. TRACKE's encoder has a branch for keyword estimation and its decoder generates captions while referring to the estimated keywords for reducing word-selection indeterminacy. In the training phase, a set of ground-truth keywords is extracted from the ground-truth caption, and the branch is trained to minimize the estimation error of the keywords. A summary of our contributions is as follows.

1. We decompose AAC into a combined task of caption generation and keyword estimation, and keyword estimation is executed by adopting a weakly supervised polyphonic AED strategy [14].
2. This is the first study that has adopted Transformer [13] to AAC[2]. We also extended Transformer to simultaneously solve the word-selection indeterminacy problem of AAC and the related AED/ASC sub-task.

2. Preliminaries of audio captioning

AAC is a task to translate an input audio sequence $(\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_T)$ into a word sequence $(w_1, ..., w_N)$. Here, $\boldsymbol{\phi}_t \in \mathbb{R}^{D_x}$ is a set of acoustic features at time index t, and T is the length of the

[1] A conventional method [4] uses ASC-aware acoustic features such as the bottleneck feature of VGGish [12]. In contrast, we attempt to solve the word-selection indeterminacy problem of AAC explicitly by using the AED/ASC sub-task.

[2] The use of a Transformer in AED/ASC tasks has been investigated [15, 16].

http://dx.doi.org/10.21437/Interspeech.2020-2087

input sequence. The output of AAC $w_n \in \mathbb{N}$ denotes the n-th word's index in the word vocabulary, and N is the length of the output sequence.

Previous studies addressed AAC using a sequence-to-sequence model (seq2seq) [17,18]. First, the encoder $\mathcal{E}$ embeds the input sequence into a feature-space as $\boldsymbol{\nu}$. Here, $\boldsymbol{\nu}$ can be either a fixed dimension vector or a hidden feature sequence. Then the decoder $\mathcal{D}$ predicts the posterior probability of the n-th word under the given input and 1st to $(n-1)$-th outputs recursively as

$$\boldsymbol{\nu} = \mathcal{E}_{\theta_e}\left(\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_T\right), \tag{1}$$

$$p(w_n|\boldsymbol{\nu}, \boldsymbol{w}_{n-1}) = \mathcal{D}_{\theta_d}\left(\boldsymbol{\nu}, \boldsymbol{w}_{n-1}\right), \tag{2}$$

where θ_e and θ_d are the sets of parameters of $\mathcal{E}$ and $\mathcal{D}$, respectively, $\boldsymbol{w}_{n-1} = (w_1, ..., w_{n-1})$, and w_n is estimated from the posterior using beam search decoding.

As mentioned above, one of the problems with AAC is indeterminacy in word selection. Since one acoustic event/scene can be described with several words, the number of possible captions becomes huge due to combinatorial explosion. To reduce such indeterminacy, previous studies used meta information such as keywords [4,5]. We define $\boldsymbol{m} = \{m_k \in \mathbb{N}\}_{k=1}^{K}$ as a set of keywords where K is the number of keywords. By passing $\boldsymbol{m}$ to the decoder, it is expected that $\boldsymbol{m}$ works as an attention factor to select the keyword from the possible words corresponding to the acoustic event/scene. Thus, (2) can be rewritten as

$$p(w_n|\boldsymbol{\nu}, \boldsymbol{m}, \boldsymbol{w}_{n-1}) = \mathcal{D}_{\theta_d}\left(\boldsymbol{\nu}, \boldsymbol{m}, \boldsymbol{w}_{n-1}\right). \tag{3}$$

3. Proposed Model

In real-world applications, there are not many use-cases for AAC systems that require keywords. If the user can input such keywords, he/she should know the sound environment without any captions. To expand the use-cases of AAC, TRACKE generates a caption while estimating its keywords from the input audio. Sections 3.1 and 3.2 give an overview and details of TRACKE, respectively, and Section 3.3 describes the procedure for extracting ground-truth keywords from the ground-truth caption.

3.1. Model overview

Figure 2 shows the architecture of TRACKE. The components of the encoder and decoder are the same of those of the original Transformer [13], but the number of stacks and hidden dimensions different. We use the bottleneck feature of VGGish [12] ($D_x = 128$) for audio embedding, and fastText [19] trained on the Common Crawl corpus ($D_w = 300$) for caption-word and keyword embedding, respectively. Since the dimension[s?] of audio feature and word embedding differ, we use two linear layers to adjust the dimensions of audio and word/keyword embedding to $D_f = 100$, which is the hidden dimension of the encoder/decoder.

In TRACKE, the size of the encoder output $\boldsymbol{\nu}$ is $D_f \times T$. The $\boldsymbol{\nu}$ is passed to the keyword-estimation branch $\mathcal{M}$ as

$$\hat{\boldsymbol{m}} = \mathcal{M}_{\theta_m}\left(\boldsymbol{\nu}\right), \tag{4}$$

where $\hat{\boldsymbol{m}} = \{\hat{m}_k \in \mathbb{N}\}_{k=1}^{K}$ is the set of the estimated keywords, and θ_m is the parameter of $\mathcal{M}$. First, to input $\hat{\boldsymbol{m}}$ to $\mathcal{D}$, $\hat{\boldsymbol{m}}$ is embedded into the feature space using fastText word embedding. To adjust the feature dimension, the embedded keywords

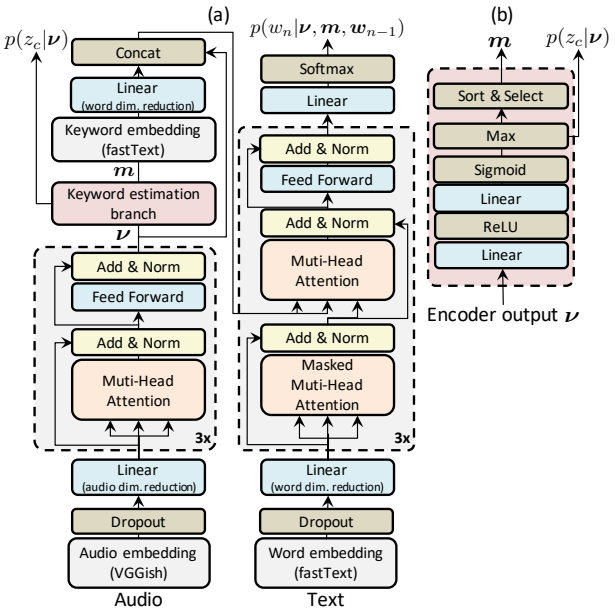

Figure 2: *(a) Architecture of TRACKE and (b) details of keyword-estimation branch $\mathcal{M}$.*

are then passed to the linear layer for dimension reduction of words/keywords. Then, the output $\mathbb{R}^{D_f \times K}$ is concatenated to $\boldsymbol{\nu}$. Finally, the concatenated feature $\mathbb{R}^{D_f \times (T+K)}$ is used as the key and value of the multi-head attention layers in $\mathcal{D}$, and the decoder estimates the posterior of the n-th word, the same as in (3), as

$$p(w_n|\boldsymbol{\nu}, \hat{\boldsymbol{m}}, \boldsymbol{w}_{n-1}) = \mathcal{D}_{\theta_d}\left(\boldsymbol{\nu}, \hat{\boldsymbol{m}}, \boldsymbol{w}_{n-1}\right). \tag{5}$$

3.2. Keyword-estimation branch

Let C be the size of the keyword vocabulary and $\boldsymbol{m}$ be a set of keywords extracted from the ground-truth caption (described in Section 3.3). The $\mathcal{M}$ estimates $\boldsymbol{m}$, that is, whether the input audio includes audio events/scenes corresponding to keywords in the keyword vocabulary.

The duration of each event/scene is different, e.g., a passing train sound is long, while a dog barking is short. Thus, as in polyphonic AED [20, 21], it would be better to estimate whether the pre-defined c-th event/scene has happened for each t. However, the given keyword labels are weak; start and stop time indexes are not given. Therefore, we carry out keyword estimation through the weakly supervised polyphonic AED strategy [14] by (i) estimating the posterior of each event on each t, $p(z_{c,t}|\boldsymbol{\nu})$, then (ii) aggregating these posteriors for all t, $p(z_c|\boldsymbol{\psi})$. Then, the most likely K events/scenes (i.e. keywords) are selected.

First is the posterior-estimation step; $\mathcal{M}$ estimates the posterior of the c-th keyword at t as

$$\hat{\boldsymbol{Z}} = \mathrm{sigmoid}\left(\mathrm{Linear}\left(\mathrm{ReLU}\left(\mathrm{Linear}\left(\boldsymbol{\nu}\right)\right)\right)\right), \tag{6}$$

where $\hat{\boldsymbol{Z}} \in [0,1]^{C \times T}$ and its (c,t) element is $p(z_{c,t}|\boldsymbol{\nu})$. Next is the posterior-aggregation step. We use the global max pooling strategy as follows because the maximum value rather than the average for considering the difference in the duration for each event

$$p(z_c|\boldsymbol{\nu}) \approx \max_t \left[p(z_{c,t}|\boldsymbol{\nu})\right]. \tag{7}$$

Then, $p(z_c|\boldsymbol{\nu})$ is sorted in descending order and the top-K keywords with high posterior are selected as $\hat{\boldsymbol{m}}$.

Note that the estimated order of the top-K keywords has no effect on text generation because position encoding is not applied to the embedding vector of $\hat{\boldsymbol{m}}$. In addition, the computational graph is not connected from $\mathcal{M}$ to $\mathcal{D}$ because the sorting and top-K selection after (7) are not differentiable. Therefore, text-generation loss is not back-propagated to two linear layers in (6), i.e. the update of θ_m is only affected by the accuracy of keyword estimation.

3.3. Rule-based keyword extraction for training

We describe a rule-based keyword extraction for generating $\boldsymbol{m}$. The keyword-estimation problem has been tackled as a sub-task of text summarization and comprehension, and several machine learning-based methods have been proposed [22–26]. In this study, the first attempt to reveal whether the use of estimated keywords is effective for AAC, we adopted a simple rule-based keyword extraction method.

We use frequent word lemmas of nouns, verbs, adjectives, and adverbs as keywords. From all captions in the training data, we first extract words that belong to the four parts of speech. Next, these words are converted to their lemmas and counted. Then, the keyword vocabulary is constructed using the most frequent C lemmas except "be". Finally, the word lemmas that exist in the keyword vocabulary are used as the ground-truth keywords $\boldsymbol{m}$. In the case of a ground-truth caption in the Clotho dataset [5] "*A muddled noise of broken channel of the TV*", the words that belong to the four target parts of speech are {muddled, noise, broken, channel, TV}. These words are then converted to their lemmas as {muddle, noise, break, channel, TV}. Finally, the lemmas that exist in the keyword vocabulary are extracted as $\boldsymbol{m}$.

3.4. Training procedure

TRACKE is trained to minimize two cost functions simultaneously; for captioning $\mathcal{L}^{\text{cap}}_{\theta_e,\theta_d}$ and keyword estimation $\mathcal{L}^{\text{key}}_{\theta_e,\theta_m}$. For $\mathcal{L}^{\text{cap}}_{\theta_e,\theta_d}$, we used the basic cross-entropy loss as $\mathcal{L}^{\text{cap}}_{\theta_e,\theta_d} = N^{-1}\sum_{n=1}^{N}\text{CE}\left(w_n, p(w_n|\boldsymbol{\nu},\hat{\boldsymbol{m}},\boldsymbol{w}_{n-1})\right)$, where CE is the cross-entropy between a given label and estimated posterior. For $\mathcal{L}^{\text{key}}_{\theta_e,\theta_m}$, to avoid $\mathcal{M}$ from always outputting the most frequent keywords, we calculate weighted binary cross-entropy, the weight of which is the reciprocal of the prior probability, as

$$\mathcal{L}^{\text{key}}_{\theta_e,\theta_m} = -\frac{1}{C}\sum_{c=1}^{C}\lambda_c z_c \ln\hat{z}_c + \gamma_c(1-z_c)\ln(1-\hat{z}_c), \quad (8)$$

where $\hat{z}_c = p(z_c|\boldsymbol{\psi})$, and $z_c = 1$ if $c \in \boldsymbol{m}$; otherwise, $z_c = 0$. Here, λ_c and γ_c are the weights as $\lambda_c = (p(z_c))^{-1}$ and $\gamma_c = (1-p(z_c))^{-1}$, respectively, where $p(z_c)$ is the prior probability of the c-th keyword calculated by

$$p(z_c) = \frac{\text{\# of } c\text{-th keyword in training captions}}{\text{\# of training captions}}. \quad (9)$$

4. Experiments

4.1. Experimental setup

Dataset and metrics: We evaluated TRACKE on the Clotho dataset [5], which consists of audio clips from the Freesound platform [27] and its captions were annotated via crowdsourcing [28]. This dataset was used in a challenge task of the Detection and Classification of Acoustic Scenes and Events (DCASE) 2020 Challenge [29]. We used the development split of 2893 audio clips with 14465 captions (i.e. one audio clip has five ground-truth captions) for training and the evaluation split of 1045 audio clips with 5225 captions for testing. From the development split, 100 audio clips and their captions were randomly selected as the validation split. We evaluated TRACKE and three other models on the same metrics used in the DCASE 2020 Challenge, i.e., BLEU-1, BLEU-2, BLEU-3, BLEU-4, ROUGE-L, METEOR, CIDEr, SPICE, and SPIDEr.

Training details: All captions were tokenized using the word tokenizer of the natural language toolkit (NLTK) [30]. All tokens in the development dataset were then counted, and words that appeared more than five times were appended in the word vocabulary. The vocabulary size was 2145, which includes BOS, EOS, PAD, and UNK tokens. The part-of-speech (POS)–tagging and lemmatization for keyword extraction were carried out using the POS–tagger and the WordNet Lemmatizer of the NLTK, respectively. Then, the most frequent $C = 50$ lemmas were appended to the keyword vocabulary. The average number of keywords per caption was 2.23, and we used $K = 5$ because the number of keywords of 95% of the training samples was less than five.

The encoder and decoder of TRACKE are composed of a stack of three identical layers, and each layer's multi-head attention/self-attention has four heads. All parameters in TRACKE were initialized using a random number from $\mathcal{N}(0, 0.02)$ [31]. The number of hidden units was $D_f = 100$, and the initial and encoder/decoder's dropout probability were 0.5 and 0.3, respectively. We used the Adam optimizer [32] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$ and varied the learning rate as the same formula of the original Transformer [13]. TRACKE was trained for 300 epochs with a batch size of 100, and the best validation model was used as the final output.

Comparison methods: TRACKE (`Ours`) was compared with three other models:

`Baseline` The baseline model of the DCASE 2020 Challenge Task 6 [1].

`LSTM` Long short-term memory (LSTM)–based seq2seq model [17, 18]. $\mathcal{E}$ was two-layer bidirectional-LSTM, and its outputs were aggregated by an attention layer. $\mathcal{D}$ was one-layer LSTM whose initial hidden state was the encoder output. The number of hidden units was 180.

`Transformer` Transformer-based AAC. Its architecture is the same as TRACKE, except that the keyword-estimation branch was removed.

To investigate the effect of the number of keywords K, we also evaluated TRACKE with $K = 10$ (`Ours`($K = 10$)), where $K = 10$ was larger than the maximum number of keywords per audio clip in the training data. To confirm the upper-bound performance of TRACKE, we also compared it with two other models. One is `Oracle1`; instead of $\hat{\boldsymbol{m}}$, the keywords in the meta-data of the Clotho dataset (i.e. Freesound tags) are passed to the decoder in both training/test stages, and the other is `Oracle2`; instead of $\hat{\boldsymbol{m}}$, all 5 ground-truth captions of $\boldsymbol{m}$ is passed to the decoder in both training/test stages. `Oracle1` gives the oracle performance when the keywords are given manually, and `Oracle2` gives this when the estimation accuracy of the keywords is perfect.

4.2. Results

Table 1 shows the evaluation results on the Clotho dataset. These results suggest the following:

Table 1: *Experimental results on Clotho dataset with DCASE2020 Challenge metrics*

Model	# of params.	B-1	B-2	B-3	B-4	CIDEr	METEOR	ROUGE-L	SPICE	SPIDEr
`Baseline`	4.64M	38.9	13.6	5.5	1.5	7.4	8.4	26.2	3.3	5.4
`LSTM`	1.12M	49.4	28.5	16.9	10.0	22.2	14.5	33.4	9.0	15.6
`Transformer`	1.11M	50.2	29.9	18.3	10.2	23.3	14.1	33.7	9.1	16.2
`Ours`$(K = 10)$	1.13M	49.9	29.7	18.4	**10.8**	23.0	14.5	**34.5**	9.1	16.1
`Ours`	1.13M	**52.1**	**30.9**	**18.8**	10.7	**25.8**	**14.9**	34.2	**9.7**	**17.7**
`Oracle1`	1.11M	53.4	32.2	20.0	11.7	27.5	15.4	35.1	10.1	18.8
`Oracle2`	1.11M	56.7	37.5	24.8	15.9	34.7	18.1	39.1	12.3	23.5

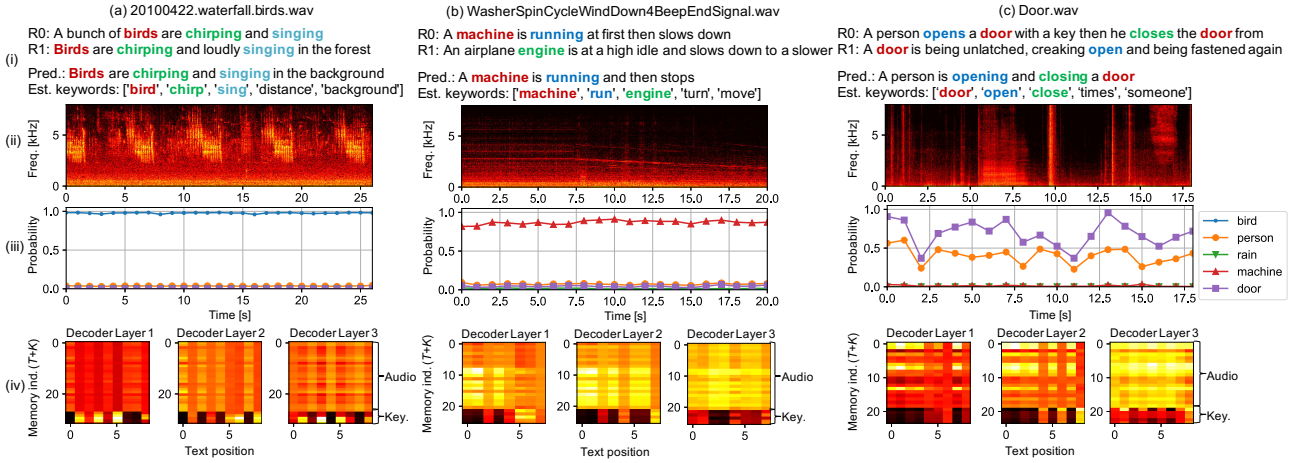

Figure 3: *Examples of TRACKE outputs. (i) Ground truth (R0 and R1) and estimated caption (Pred.) and keywords (Est. keywords), (ii) input spectrogram, (iii) keyword posterior of each time index $p(z_{c,t}|\psi)$, and (iv) attention matrices of decoder.*

(i) $\mathcal{M}$ works effectively for AAC. TRACKE (`Ours`) achieved the highest score without given keywords. In addition, the BLEU-1 (ROUGE-L) score of `Ours` was 52.1 (34.2), while that of `Oracle1`, which uses manually given keywords, was 53.4 (35.1). Thus, the score of `Ours` was 97.6% (97.4%) compared with `Oracle1`, in spite the fact that `Ours` is a perfectly automated audio-captioning model.

(ii) If TRACKE can accurately estimate the keywords, performance might further improve. The oracle performance of `Oracle2` was significantly higher than that of `Ours`. Since the keyword estimation accuracy of `Ours` was 48.1%[3], we need to improve this in future work.

(iii) If the estimated K is too large, the use of the estimated keywords in text generation might be ineffective in reducing indeterminacy in word selection because the scores of `Transformer` and `Ours`$(K = 10)$ were almost the same. To further improve the performance of TRACKE, K should also be estimated from the input.

(iv) Transformer might be effective for AAC because `Transformer` was slightly better than `LSTM`. However, since the training of Transformer requires a large-scale dataset, to affirm the effectiveness of Transformer, we need to evaluate Transformer by developing more large-scale datasets for AAC[4].

(v) The use of pre-trained models is effective because there were large performance gaps between `Baseline` and the others, and the major difference was the use of pre-trained models such as VGGish [12] and fastText [19].

Figure 3 shows examples of TRACKE outputs. These results suggest that indeterminacy words were determined while referring to the estimated keywords, for example, (b) {machine, airplane} and (c) {close, fasten}. In addition, the posterior probabilities of keywords imply the implicit co-occurrence relationships, rather than just classifying acoustic events/scenes. In (c), the posterior probability of "person" increased even though human sounds, such as speech, were not included in the input audio. This might be the result of exploiting the co-occurrence relationship that opening and closing a door is usually done by humans.

5. Conclusions

We proposed a Transformer-based audio captioning model with keyword estimation called *TRACKE*, which simultaneously solves the word-selection indeterminacy problem of the main task of ACC while executing the AED/ASC- sub-task (i.e. keyword estimation). TRACKE estimates the keywords of the target caption from input audio, and its decoder generates a caption while referring to the estimated keywords. The keyword-estimation branch was trained by adopting a weakly supervised polyphonic AED strategy [14], and the ground-truth keywords were extracted from the ground-truth caption via a heuristic rule. The experimental results indicate the effectiveness of TRACKE for AAC.

Future work includes improving keyword estimation while adopting keyword-guided generation strategies in natural language processing [23–25, 33, 34] and image captioning [35–38].

[3]The percentage of estimated keywords that were included in the ground-truth keywords.

[4]The number of training sentence pairs in natural language processing datasets, such as WMT 2014 English-French dataset, for machine translation is 36 million.

6. References

[1] K. Drossos, S. Adavanne, and T. Virtanen, "Automated Audio Captioning with Recurrent Neural Networks," in *Proc. of IEEE Workshop on Application of Signal Process. to Audio and Acoust. (WASPAA)*, 2017.

[2] S. Ikawa and K. Kashino, "Neural Audio Captioning based on Conditional Sequence-to-Sequence Model," in *Proc. of the Detection and Classification of Acoust. Scenes and Events Workshop (DCASE)*, 2019.

[3] M. Wu, H. Dinkel, and K. Yu, "Audio Caption: Listen and Tell," in *Proc. of Int'l Conf. on Acoust., Speech, and Signal Process. (ICASSP)*, 2019.

[4] C. D. Kim, B. Kim, H. Lee, and G. Kim, "AudioCaps: Generating Captions for Audios in The Wild," in *Proc. of the North American Chapter of the Association for Computational Linguistics: Human Lang. Tech. (NAACL-HLT)*, 2019.

[5] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: An Audio Captioning Dataset," in *Proc. of Int'l Conf. on Acoust., Speech, and Signal Process. (ICASSP)*, 2020.

[6] Y. Koizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "The NTT DCASE2020 Challenge Task 6 System: Automated Audio Captioning with Keywords and Sentence Length Estimation," in *Tech. Report of the Detection and Classification of Acoust. Scenes and Events Workshop (DCASE) Challenge*, 2020.

[7] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen, "Acoustic Event Detection in Real Life Recordings," in *Proc. of Euro. Signal Process. Conf. (EUSIPCO)*, 2010.

[8] K. Imoto, N. Tonami, Y. Koizumi, M. Yasuda, R. Yamanishi, and Y. Yamashita, "Sound Event Detection By Multitask Learning of Sound Events and Scenes with Soft Scene Labels," in *Proc. of Int'l Conf. on Acoust., Speech, and Signal Process. (ICASSP)*, 2020.

[9] D. Barchiesi, D. Giannoulis, D. Stowell, and M. D. Plumbley, "Acoustic Scene Classification: Classifying Environments from the Sounds they Produce," *IEEE Signal Processing Magazine*, 2015.

[10] Y. Koizumi, S. Saito, H. Uematsu, Y. Kawachi, and N. Harada, "Unsupervised Detection of Anomalous Sound based on Deep Learning and the Neyman-Pearson Lemma," *IEEE/ACM Tran. on Audio, Speech, and Lang. Process.*, 2019.

[11] J. F. Gemmeke, D. P. W. 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. of Int'l Conf. on Acoust., Speech, and Signal Process. (ICASSP)*, 2017.

[12] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, DvPlatt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "CNN Architectures for LargeScale Audio Classification," in *Proc. of Int'l Conf. on Acoust., Speech, and Signal Process. (ICASSP)*, 2017.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Proc. of Neural Information Processing Systems (NIPS)*, 2017.

[14] R. Serizel, N. Turpault, H. E. Zadeh, and A. P. Shah, "Large-Scale Weakly Labeled Semi-Supervised Sound Event Detection in Domestic Environments," in *Proc. of the Detection and Classification of Acoust. Scenes and Events Workshop (DCASE)*, 2018.

[15] W. Boes and H. V. Hamme, "Audiovisual Transformer Architectures for Large-Scale Classification and Synchronization of Weakly Labeled Audio Events," in *Proc. of Int'l Conf. on Multimedia (MM'19)*, 2019.

[16] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, "Sound Event Detection of Weakly Labelled Data with CNN-Transformer and Automatic Threshold Optimization," *arXiv preprint, arXiv:1912.04761*, 2019.

[17] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Proc. of Advances in Neural Information Process. Systems (NIPS),* 2014.

[18] M. T. Luong, H. Pham, and C. D. Manning "Effective Approaches to Attention-based Neural Machine Translation," in *Proc. of Empirical Methods in Natural Lang. Process. (EMNLP),* 2015.

[19] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," *Transactions of the Association for Computational Linguistics*, 2017.

[20] E. Cakir, T. Heittola, H. Huttunen, and T. Virtanen, "Polyphonic Sound Event Detection using Multi Label Deep Neural Networks in *Int'l Joint Conf. on Neural Networks (IJCNN)*, 2015.

[21] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for Polyphonic Sound Event Detection," *Applied Sciences*, 2016.

[22] R. Mihalcea and P. Tarau, "TextRank: Bringing Order into Texts," in *Proc. of Empirical Methods in Natural Lang. Process. (EMNLP)*, 2004.

[23] C. Li, W. Xu, S. Li, and S. Gao, "Guiding Generation for Abstractive Text Summarization Based on Key Information Guide Network," in *Proc. of the North American Chapter of the Association for Computational Linguistics: Human Lang. Tech. (NAACL-HLT)*, 2018.

[24] R. Pasunuru and M. Bansal, "Multi-Reward Reinforced Summarization with Saliency and Entailment," in *Proc. of the North American Chapter of the Association for Computational Linguistics: Human Lang. Tech. (NAACL-HLT)*, 2018.

[25] S. Gehrmann, Y. Deng, and A. M. Rush, "Bottom-up Abstractive Summarization," in *Proc. of Empirical Methods in Natural Lang. Process. (EMNLP)*, 2018.

[26] K. Nishida, I. Saito, K. Nishida, K. Shinoda, A. Otsuka, H. Asano and J. Tomita, "Multi-style Generative Reading Comprehension," in *Proc. of the 57th Ann. Meet. of the Association for Computational Linguistics (ACL 2019)*, 2019.

[27] F. Font, G. Roma, and X. Serra, "Freesound Technical Demo," in *Proc. of Int'l Conf. on Multimedia (MM'13)*, 2013.

[28] S. Lipping, K. Drossos, and T. Virtanen, "Crowdsourcing a Dataset of Audio Captions," in *Proc. of the Detection and Classification of Acoust. Scenes and Events Workshop (DCASE)*, 2019.

[29] "DCASE2020 Challenge Task 6: Automated Audio Captioning," `http://dcase.community/challenge2020/task-automatic-audio-captioning`

[30] S. Bird, E. Loper and E. Klein, "Natural Language Processing with Python," *O'Reilly Media Inc.*, 2009.

[31] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving Language Understanding by Generative Pre-Training," `https://blog.openai.com/language-unsupervised`, 2018.

[32] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. of Int.l Conf. Learn. Representations (ICLR)*, 2015.

[33] I. Saito, K. Nishida, K. Nishida, and J. Tomita, "Abstractive Summarization with Combination of Pre-trained Sequence-to-Sequence and Saliency Models", *arXiv preprint, arXiv:2003.13028*, 2020.

[34] I. Saito, K. Nishida, K. Nishida, A. Otsuka, H. Asano, and J. Tomita, "Length-controllable Abstractive Summarization by Guiding with Summary Prototype", *arXiv preprint, arXiv:2001.07331*, 2020.

[35] T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei, "Boosting Image Captioning with Attributes," in *Proc. of Int'l. Conf. on Computer Vision (ICCV)*, 2017.

[36] Y. Pan, T. Yao, H. Li, and T. Mei, "Video Captioning with Transferred Semantic Attributes," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[37] J. Lu, C. Xiong, D. Parikh, and R. Socher, "Knowing When to Look: Adaptive Attention via A Visual Sentinel for Image Captioning," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[38] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.