CP-JKU'S SUBMISSION TO TASK 6A OF THE DCASE2022 CHALLENGE: A BART ENCODER-DECODER FOR AUTOMATIC AUDIO CAPTIONING TRAINED VIA THE REINFORCE ALGORITHM AND TRANSFER LEARNING

Technical Report

Paul Primus[1], *Gerhard Widmer*[1,2]

[1]Institute of Computational Perception (CP-JKU)
[2]LIT Artificial Intelligence Lab
Johannes Kepler University, Austria

ABSTRACT

This technical report details the CP-JKU submission to the automatic audio captioning task of the 2022's DCASE challenge (task 6a). The objective of the task was to train a sequence-to-sequence model that automatically generates textual descriptions for given audio recordings. The approach described in this report enhances the BART-based encoder-decoder model used as the challenge's baseline system in three directions: firstly, the VGGish embedding model was replaced with a custom CNN10-like model that we pre-trained on AudioSet. Secondly, the BART encoder-decoder model was pre-trained on AudioCaps, which led to faster convergence. And finally, the best model was further fine-tuned by optimizing the non-differentiable CIDEr metric using the REINFORCE algorithm. Our best model achieves a SPIDEr score of .29 (single-model performance), which is an improvement of 6.6 pp. over the challenge's baseline score.

***Index Terms*—** Automatic Audio Captioning, Transfer Learning, REINFORCE

1. TASK DESCRIPTION

Automatic audio captioning [1] aims at generating whole sentences descriptions (captions) for audio signals, which convey information about the involved sounds, objects, and actions (e.g., 'A shrill squeak uttered by a male person.'), and high-level information like the temporal composition of the acoustic events (e.g., 'A car honking three times.'). Automatically creating captions is arguably more complex than basic intelligent audio processing tasks like classification and tagging because the number of potential outputs grows combinatorically with sentence length and multiple valid prediction targets for the same input exist.

In our submission, we focused on enhancing the official baseline system with three improvements that resulted in performance gains in the previous editions of the DCASE Challenge: A stronger audio embedding model pre-trained on AudioSet [2], transfer learning using the AudioCaps data set [3], and the REINFORCE algorithm [4] to directly optimize the CIDEr metric used for ranking submissions on the leader board. The resulting captioning model achieves a SPIDEr score [5] of .29 on the public test set, which is an absolute improvement of 6.6 pp. over the baseline system.

2. SYSTEM CHARACTERISTICS

The following section gives an overview of the architecture and input features.

2.1. Model Architecture

Our model's architecture (Fig. 1) is analogous to that of the baseline system, which first embeds the audio signal with the VGGish [6] network to obtain a sequence of vectors and then transforms this sequence into a textual description using a BART-like model [7]. We replaced the VGGish network with a CNN10-like [8] 10-layer convolutional neural network (called CNN10 from here on) because its representation quality proved superior in multiple studies (e.g., [8]). The architecture of CNN10 is detailed in Table 1. CNN10 outputs a sequence of 512-dimensional embedding vectors, which are converted to 768-dimensional BART encoder inputs tokens using an affine linear transformation. Both encoder and decoder consist of 6 transformer layers, with twelve attention heads in the attention layers and 3072 hidden units in feed-forward layers. The auto-regressive decoder is conditioned on the encoder output by utilizing one additional encoder-attention block after each self-attention block. The word inputs to the decoder are converted to 768-dimensional tokens using a frozen word embedding layer. The parameters of this layer were initialized by transferring the weights of a pre-trained Word2Vec Skip-Gram model [9]. The decoder output tokens are converted back to a distribution over words using a linear layer and a softmax activation. A single custom BART architecture for audio captioning, including the audio embedding network, has approximately 110 million parameters.

2.2. Audio Features

The 10-30 second long audio recordings sampled at 32kHz are converted to 64-bin log-MEL spectrograms using a 1024-point FFT with a window length of 800 (25ms) and hop size of 320 (10ms). The audio features are normalized via batch normalization [10] before feeding them into the CNN10 embedding model.

2.3. Vocabulary & Word Features

Input sentences are converted into a sequence of tokens, such that each token represents a word. Each sentence is pre-processed by converting all characters to lower case and removing punctuation. Splitting of words is done based on the white spaces between words.

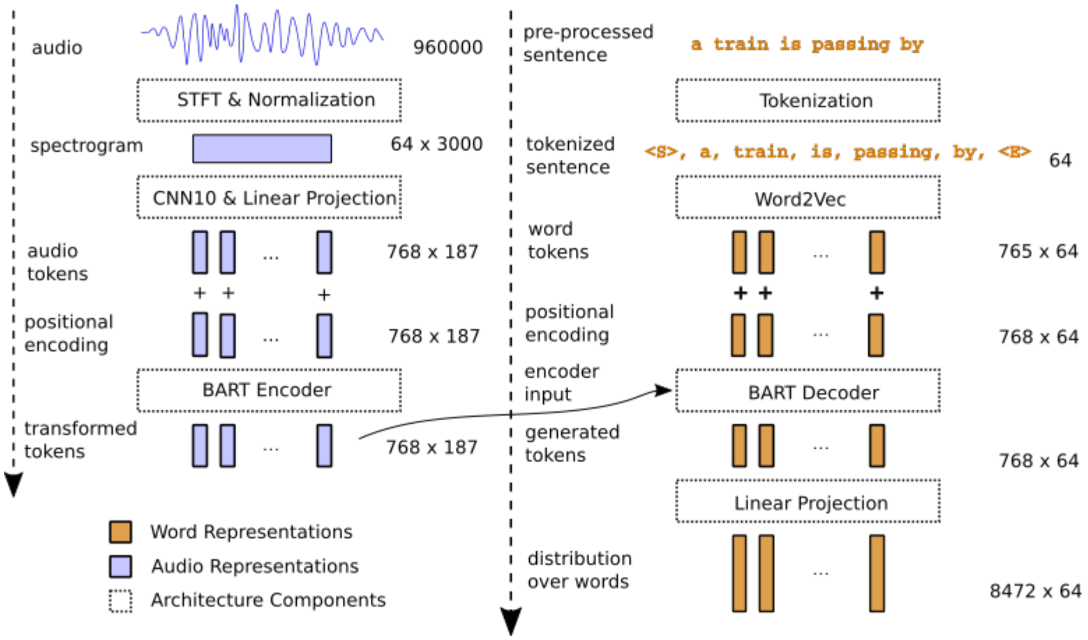

Figure 1: The architecture of the proposed approach. The numbers next to audio/ word representations are the corresponding tensor shapes; the last dimension always corresponds to time.

The resulting vocabulary includes 8472 distinct words (including special tokens) and covers all terms used in the training set.

3. TRAINING

Training of the audio embedding model and the captioning model was done in four steps, two of which are optional:

Audio Embedding (I) In the first step, the custom CNN10 audio embedding network was pre-trained on AudioSet [2], a data set designed for audio tagging that holds approximately 2 million examples labeled for 527 classes. CNN10 was initialized with parameters obtained from ImageNet [11] training. The network was then trained to minimize the binary cross-entropy between the predicted and actual labels with a constant learning rate of 10^{-3} using Adam optimizer [12] for 200 epochs; hyper-parameters were set to PyTorch's [13] defaults. To reduce overfitting and improve generalization, we used MixUp [14] on raw audios and spectrograms ($\alpha = 0.1$ and $\alpha = 1$, respectively), SpecAugment [15] ($f = 16, t = 32$), and gain augmentation ($\pm 4db$). The resulting model reaches a mean average precision of 39.84% on AudioSet's test set, which is slightly better than the 38% reported by Kong et al. [8]. To obtain the sequence of embeddings, the output of the last convolutional block of this model was averaged over the frequency dimension and transformed via a time-shared, fully-connected layer. The unused classification head was discarded after pre-training, and the embedding model's parameters were frozen for all experiments.

Pretraining on AudioCaps (II, optional) In a optional second step, the encoder-decoder model was pre-trained on the AudioCaps [3] data set for 50 epochs using Adam optimizer with a learning rate of 10^{-5} to minimize the cross-entropy loss between the predicted and the expected words w_t^* in the ground truth sentence $s^* = (w_1^*, \dots w_T^*)$ of length T:

$$\mathcal{L}_{CE} = -\frac{1}{T}\sum_{t=1}^{T} log(p(w_i^* \mid \theta)) \tag{1}$$

Training on ClothoV2 (III) In step three, the BART model was trained to minimize Eq. 1 on the ClothoV2 data set [16]. If pretraining on AudioCaps (step II) was performed before, the model was only trained for 30 additional epochs. In this case, the learning rate started at 10^{-5} and was linearly decayed to 5×10^{-6} from epoch 20 onward. If no pre-training on AudioCaps was performed, the model was trained from scratch for 60 epochs. The initial learning was set to 10^{-5}, which was linearly decayed to 5×10^{-6} in between epoch 20 and 30 and kept constant after that.

REINFORCE (IV, optional) In an optional fourth step, the resulting model was further fine-tuned for 60 more epochs by minimizing the expected value of the non-differentiable CIDERr score as described by Mei et al. [17]:

$$\mathcal{L}_{score} = -\mathbb{E}_{s \sim p(s|\theta)}\left[\text{CIDEr}(s, s^*)\right] \tag{2}$$

$s = (w_1, \dots, w_T)$ is a sentence sampled in normal mode (i.e., no teacher forcing), and $\text{CIDEr}(s, s^*)$ is the CIDEr score between the predicted sentence s and the ground truth sentence s^*. The score function estimator gives the gradient wrt. to Eq. 2:

$$\nabla_\theta \mathcal{L}_{score} = -\mathbb{E}_{s \sim p(s|\theta)}\left[\text{CIDEr}(s, s^*)\nabla_\theta \log p(s \mid \theta)\right] \tag{3}$$

This expectation was approximated via a single Monte Carlo sample and the CIDEr score between the greedily decoded sentence s_g and the ground truth was used as a baseline to reduce the variance:

$$\nabla_\theta \mathcal{L}_{score} \approx \left(\text{CIDEr}(s_g, s^*) - \text{CIDEr}(s, s^*)\right)\nabla_\theta \log p(s \mid \theta) \tag{4}$$

CNN10
$2 \times (3 \times 3)$@64, BN, ReLU
Pool (2×2)
$2 \times (3 \times 3)$@128, BN, ReLU
Pool (2×2)
$2 \times (3 \times 3)$@256, BN, ReLU
Pool (2×2)
$2 \times (3 \times 3)$@512, BN, ReLU
Pool (2×2)
Frequency Pooling
FC 512, ReLU (shared over time)
FC 527, Sigmoid (classificaiton)

Table 1: The architecture of the audio embedding model (CNN10). The embedding vectors that are produced by the first Fully-Connected (FC) layer are used as input to the BART encoder.

The Adam optimizer was used to minimize Eq. 2 with estimated gradients from Eq. 4. The learning rate was initially set to 10^{-5}, linearly decayed to 5×10^{-6} in between epochs 20 and 30, and finally kept constant for the remaining 30 epochs. SpecAugment was utilized with the previously reported hyper-parameters in all training stages to reduce overfitting on the audio inputs.

4. RESULTS

The evaluation results of models trained with variants of the previously described procedure are given in Table 2. The results suggest that all three introduced modifications (using CNN10 embeddings, pre-training on AudioCaps, and fine-tuning with the REINFORCE algorithm) lead to an improvement over the baseline system.

We submit predictions of following systems to the challenge:

- **Submission 1**: A single BART model trained on ClothoV2 only (step I & III).
- **Submission 2**: A single BART model pre-trained on AudioCaps and fine tuned on ClothoV2 (step I-III).
- **Submission 3**: Submission 2 fine-tuned for 60 more epochs using the REINFORCE algorithm (step I-IV).
- **Submission 4**: An ensemble of the 6 best BART models checkpoints of submissions 2 & 3. Ensembling was done during decoding by selecting the next word in each step based on the average value of the unnormalized outputs of all models.

5. ACKNOWLEDGMENT

The LIT AI Lab is financed by the Federal State of Upper Austria.

6. REFERENCES

[1] K. Drossos, S. Adavanne, and T. Virtanen, "Automated audio captioning with recurrent neural networks," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, WASPAA*. IEEE, 2017.

[2] 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 *2017 IEEE Int. Conf. on Acoustics, Speech and Signal Processing, ICASSP*, 2017.

[3] C. D. Kim, B. Kim, H. Lee, and G. Kim, "AudioCaps: Generating captions for audios in the wild," in *Proc. of the North American Ch. of the Ass. for Computational Linguistics: Human Language Technologies, NAACL-HLT*, 2019.

[4] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, 1992.

[5] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, "Improved image captioning via policy gradient optimization of spider," in *IEEE International Conference on Computer Vision, ICCV*, 2017.

[6] 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. W. Wilson, "CNN architectures for large-scale audio classification," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process., ICASSP*, 2017.

[7] 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. of the 58th Annual Meeting of the Association for Computational Linguistics, ACL*, 2020.

[8] 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.*, 2020.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *1st Int. Conf. on Learning Representations, ICLR*, 2013.

[10] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. of the 32nd Int. Conf. on Machine Learning, ICML*, 2015.

[11] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition CVPR*, 2009.

[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd Int. Conf. on Learning Representations, ICLR*, 2015.

[13] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Annual Conf. on Neural Information Processing Systems, NEURIPS*, 2019.

[14] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *6th Int. Conf. on Learning Representations, ICLR*, 2018.

Steps	BELU$_1$	BELU$_2$	BELU$_3$	BELU$_4$	METEOR	ROUGE$_L$	CIDERr	SPICE	SPIDEr
DCASE baseline	.555	.358	.239	.156	.164	.364	.358	.109	.233
I & III	.566	.374	.252	.164	.178	.376	.408	.120	.264
I - III	.573	.370	.245	.158	.181	.376	.440	.128	.284
I - IV	**.653**	**.424**	**.278**	.169	.181	**.404**	.455	.125	.290
Ensemble	.637	.417	.275	.168	**.183**	.401	**.460**	**.130**	**.295**

Table 2: Results on the ClothoV2 test set. Step I: Audio Embedding, Step II: Pretraining on AudioCaps, Step III: Training on ClothoV2 , Step IV: CIDEr optimization with REINFORCE

[15] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, “SpecAugment: A simple data augmentation method for automatic speech recognition,” in *20th Annual Conf. of the Int. Speech Communication Association, Interspeech*, 2019.

[16] K. Drossos, S. Lipping, and T. Virtanen, “Clotho: an audio captioning dataset,” in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process., ICASSP*, 2020.

[17] X. Mei, Q. Huang, X. Liu, G. Chen, J. Wu, Y. Wu, J. Zhao, S. Li, T. Ko, H. Tang, X. Shao, M. D. Plumbley, and W. Wang, “An encoder-decoder based audio captioning system with transfer and reinforcement learning,” in *Proc. of the 6th Workshop on Detection and Classification of Acoustic Scenes and Events, DCASE*, 2021.