IMPROVING NATURAL-LANGUAGE-BASED AUDIO RETRIEVAL WITH TRANSFER LEARNING AND AUDIO & TEXT AUGMENTATIONS

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

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

ABSTRACT

The absence of large labeled datasets remains a significant challenge in many application areas of deep learning. Researchers and practitioners typically resort to transfer learning and data augmentation to alleviate this issue. We study these strategies in the context of audio retrieval with natural language queries (Task 6b of the DCASE 2022 Challenge). Our proposed system uses pretrained embedding models to project recordings and textual descriptions into a shared audio-caption space in which related examples from different modalities are close. We employ various data augmentation techniques on audio and text inputs and systematically tune their corresponding hyperparameters with sequential model-based optimization. Our results show that the used augmentations strategies reduce overfitting and improve retrieval performance.

Index Terms— Language-based Audio Retrieval, Transfer Learning, Audio Augmentation, Text Augmentation

1. INTRODUCTION

Natural-language-based audio retrieval is concerned with ranking audio recordings depending on their content's similarity to textual descriptions. Retrieval tasks like this are typically solved by converting recordings and textual descriptions into high-level representations and then aligning them in a shared audio-caption space; ranking can then be done based on the distance between embeddings. These systems' retrieval performance highly depends on the quality of the audio and text embedding models, which must extract features that accurately and discriminatively represent the high-level content. Current state-of-the-art approaches [1, 2, 3] create such feature extractors by training models with millions of parameters directly from raw input features, i.e., deep learning. These large embedding models require a large number of training examples, such as the 400 million image-text pairs used to train CLIP [4], a cutting-edge image-retrieval model. However, publicly available audio-caption datasets like Clotho and AudioCaps are significantly smaller. This work showcases how to use off-the-shelf pretrained audio and text neural networks to create a state-of-the-art retrieval model under this limiting condition. We evaluate our approach in the context of task 6b[1] of the 2022's DCASE Challenge [5], which is concerned with audio retrieval from natural language descriptions. We demonstrate how an already well-performing baseline model can be further improved by using a range of audio and text augmentation methods and pretraining on AudioSet.

[1]https://dcase.community/challenge2022/task-language-based-audio-retrieval

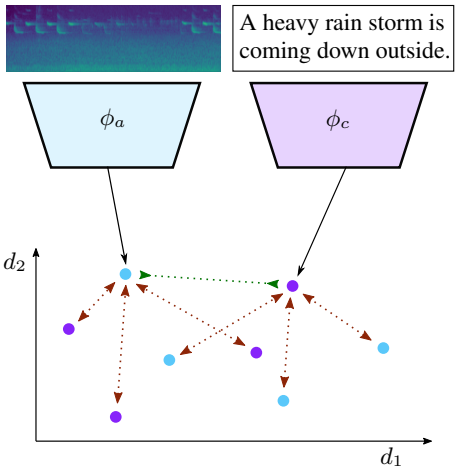

Figure 1: The proposed audio-retrieval system in a nutshell: Audio and descriptions are transformed into the shared audio-caption embedding space via the audio and description embedding models ϕ_a and ϕ_c, respectively. The contrastive loss maximizes the similarities between matching pairs.

2. RELATED WORK

The idea of aligning text and audio features for content-based retrieval is not new: Early audio retrieval methods connected bag-of-words text queries and MFCC features via density or discriminative models [6]. However, the handcrafted features and the relatively small vocabulary limited these methods' performance. Current methods build on top of learnable feature extractors that produce high-level audio and text representations from raw input features. Xie et al. [2], for example, used a convolutional recurrent neural network to extract frame-wise acoustic embeddings and aligned those to Word2Vec features via a linear transformation. Recently, language-based audio retrieval has received increased attention due to the newly introduced task 6b in the 2022's DCASE challenge [5]. The task's objective was to create a retrieval system that takes natural-language queries as input and retrieves the ten best-matching recordings from a test set. The top ranking systems among the nine submitted ones leveraged large pretrained audio and text embedding models like CNN14 [7] and BERT [8], respectively. While most systems applied SpecAugment [9], other data augmentation methods, especially text augmentations, have received little to no attention. We address this paucity and study a range of audio and text augmentation methods in the context of audio retrieval.

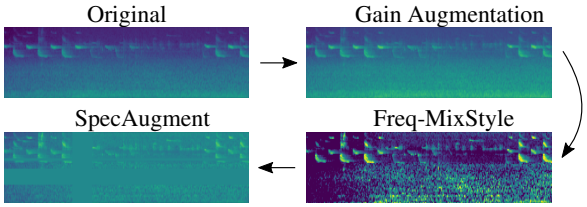

Figure 2: Overview of the audio augmentation pipeline.

Augmentation	Caption
Original	The rain pours down.
Back Translation	It rains cats and dogs.
Insert	It tree rains cats and dogs.
Delete	It rains cats and ~~dogs~~.
Swap	It and cats rains dogs.
Synonym	It drizzles cats and dogs.

Table 1: Overview of the text augmentation pipeline

3. RETRIEVAL SYSTEM

Our model uses separate audio and caption embedding networks $\phi_a(\cdot)$ and $\phi_c(\cdot)$ to embed tuples of spectrograms and descriptions $\{(a_i, c_i)\}_{i=1}^{N}$ into a shared D-dimensional space in a manner that representations of matching audio-caption pairs are close. This behavior is achieved by contrastive training, which equalizes the embeddings of matching audio-caption pairs (a_i, c_i), while pushing the representations of mismatching pairs $(a_i, c_{j;j\neq i})$ apart. The agreement between audio a_i and description c_j is estimated via the normalized dot product in the shared embedding space:

$$C_{ij} = \frac{\phi_{\mathrm{a}}(a_i)^T \cdot \phi_{\mathrm{c}}(c_j)}{\|\phi_{\mathrm{a}}(a_i)\|^2 \, \|\phi_{\mathrm{t}}(c_j)\|^2}$$

The similarity matrix $\mathbf{C} \in \mathbb{R}^{N\times N}$ holds the agreement of matching pairs on the diagonal and the agreement of mismatching pairs off-diagonal. We train the system using the NT-Xent [10] loss, which is defined as the average Cross Entropy (CE) loss over the audio and text dimension; the ground truth is given by the identity matrix $\mathbf{I} \in \mathbb{R}^{N\times N}$:

$$\mathcal{L} = \frac{1}{2\cdot N}\sum_{i=1}^{N} \mathrm{CE}(\mathbf{C}_{i*}, \mathbf{I}_{i*}) + \mathrm{CE}(\mathbf{C}_{*i}, \mathbf{I}_{*i})$$

4. AUDIO AUGMENTATIONS

To reduce overfitting of the audio embedding model and improve generalization, we employ three regularization techniques during training: Gain augmentation, MixStyle [11] along the frequency dimension (Freq-MixStype), and SpecAugment [9]. Figure 2 gives an overview of the audio augmentation pipeline.

Gain Augmentation tries to make the model invariant changes in volume by randomly altering the loudness of the raw audio input signal. Volume manipulations are done by multiplying the waveform with factor W:

$$w = 10^{(g/20)}$$

The change in volume (in dB) is controlled with hyperparameter g; its value is randomly drawn from a uniform distribution in the range $[-g_{\mathrm{max}}, g_{\mathrm{max}}]$.

SpecAugment [9] randomly masks time and frequency stripes in the input spectrogram, thereby reducing the audio embedding model's reliance on specific input patterns. The number of stripes along the time and frequency dimensions is controlled via hyperparameters n_f and n_t, respectively. Parameters w_f and w_t control the maximum width of the time and frequency stripes, respectively. The actual width and the offset of the stripes are chosen from a uniform distribution; masked values are replaced with zeros. We omitted the warping transformation proposed in the original work as it is computationally expensive and reportedly only lead to marginal improvements.

Freq-MixStyle [11] aims to transfer device-style characteristics between recordings by exchanging statistics along the frequency dimension of spectrograms. To this end, the original spectrogram is first normalized to zero mean unit variance along the frequency dimension and then un-normalized with adjusted mean and standard deviation statistics. The adjusted statistics are a convex combination of the original statistics $(\mu_i, \sigma - i)$ and the statistic of a randomly selected spectrogram (μ_j, σ_j):

$$\mu_{\mathrm{new}} = \lambda\mu_i + (1-\lambda)\mu_j$$

$$\sigma_{\mathrm{new}} = \lambda\sigma_i + (1-\lambda)\sigma_j$$

The coefficient λ is drawn from a symmetric beta distribution in a manner that the original statistics always receive a higher weight:

$$\lambda \sim \mathrm{Beta}(\alpha, \alpha)$$

α controls the shape of the Beta distribution. Freq-MixStyle is applied to each input example with a probability of p_{MS}.

5. TEXT AUGMENTATIONS

We apply Back Translation [12] and Easy Data Augmentation [13] (in that order) to reduce overfitting of the sentence embedding model. Examples of these augmentations are given in Table 1.

Back Translation (BT) [12] introduces variation into the input sentence without changing its semantics by translating the input sentence to a foreign language and back to the source language. We translate the training captions from English to German, French, or Spanish, and back to English using Google Translate.

Easy Data Augmentation (EDA) [13] chooses one of four word-level manipulations and applies the selected operation to each word with a certain probability (indicated in parenthesis): insertion of a random word (p_{ins}), deletion (p_{del}), swap with another word in the sentence (p_{swp}), or replacement with a synonym according to WordNet [14] (p_{syn}). EDA is applied with a probability of p_{EDA}.

6. EXPERIMENTS

We first established a baseline without augmentation and then conducted a series of experiments to investigate the impact of using pretrained weights for the audio embedding model, augmenting the

audio and text inputs, and pretraining on AudioCaps. We further investigate the impact of all augmentation methods separately in an ablation study. The model architecture and the exact experimental setup are discussed below.

6.1. Dataset & Input Features

We trained our proposed system on ClothoV2.1 [15], which contains 10-30 second long audio recordings sampled at 32kHz and five human-generated captions for each recording. We used the training, validation, and test split into 3839, 1045, and 1045 examples, respectively, as suggested by the dataset's creators. To make processing in batches easier, we zero-padded all audio snippets to the maximum audio length in the batch. The resulting waveforms were converted to 64-bin log-MEL spectrograms using a 1024-point FFT (32ms) and hop size of 320 (10ms). The audio features were normalized via batch normalization [16] along the frequency dimension before feeding them into the CNN10 embedding model. The input sentences were pre-processed by converting all characters to lowercase and removing punctuation. The resulting strings were tokenized with the WordPiece tokenizer [17], padded to the maximum sequence length in the batch, and truncated to 32 tokens.

6.2. Audio Embedding Model

We used a slightly modified version of the popular CNN10 architecture [7] to embed spectrograms into the 1024-dimensional audio-caption space. The architecture is detailed in Table 2. The network aggregates the output after the last convolutional block over the frequency and time dimensions and transforms the result with a two-layer neural network. The audio embedding model has approximately 9 Million parameters. We chose this simple architecture because it allowed us to train on a single customer-grade GPU with reasonable batch size. For the experiments with pretrained audio embedding model parameters, we transferred the weights of the convolutional blocks from a custom pretrained audio tagger and randomly initialized the fully-connected layers. The data set used for pretraining the embedding model, AudioSet [18], contains approximately 2 Million ten-second audio recordings labeled for 527 hierarchically organized classes. We used the pre-defined split of AudioSet into a large, unbalanced set for training and two smaller, more balanced sets for validation and testing. Pretraining of the embedding model was done as described in [7].

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 (mean)
Time Pooling (average of mean and max)
FC 2048, ReLU
FC 1024

Table 2: The architecture of the audio embedding model (CNN10).

6.3. Text Embedding Model

We used a pretrained BERT model [8] ('bert-base-uncased') to generate embeddings for the audio captions. BERT is a bi-directional self-attention-based sentence encoder that was pretrained on BookCorpus [19] and WikiText datasets [20] for masked language modeling and next sentence prediction. The learned semantic representations proved effective in multiple downstream tasks. We projected the output vector that corresponds to the class token into the shared audio-caption space by using a neural network with one hidden layer of size 2048 and ReLu activations. The text embedding model has approximately 112 Million parameters.

6.4. Training & Evaluation

We train all variants of the proposed system on CLothoV2.1's training set, select hyperparameters according to the performance on the validation set, and report the final results on the test set in section 7. Our main evaluation criterion was the mean Average Precision among the top-10 results (mAP) because this criterion takes the rank of the correct recording into account. We also report the recall among the top-1, top-5, and top-10 retrieved results. All results are averaged over three runs. Both embedding models were jointly optimized using gradient descent with a batch size of 30. We used the Adam update rule [21] for 50 epochs, set the initial learning rate to 10^{-4}, and dropped it by a factor of 3 every 10 epochs. The hyperparameters of the optimizer were set to PyTorch's [22] defaults.

6.5. Sequential Model-Based Optimization

We performed sequential model-based optimization (SMBO) in the hyperparameter space of the audio and text augmentations to optimize the mAP-score on the validation set without manual tuning. Sequential Model-based Optimization (SMBO) utilizes the outcomes of prior experiments to build a surrogate model that estimates the relationship between validation-mAP and a given parameter configuration. Subsequent runs use this surrogate model to sample hyperparameter configurations from a distribution that is proportional to the expected mAP improvement. We initialized SMBO with ten runs using randomly chosen hyperparameters. After that, we performed 100 trials with hyperparameters sampled using the Tree-structured Parzen Estimator algorithm [23]. To reduce the overall computation time, we stopped runs for which the mAP on the validation set did not increase for ten consecutive epochs. Table 4 defines the hyperparameter search space for the SMBO.

7. RESULTS & DISCUSSION

The results of our experiments are summarized in Table 3 and discussed in the following section.

	R@1	R@5	R@10	mAP@10
DCASE baseline	3.50	11.50	19.50	7.50 ± 0.00
baseline	6.63	20.06	31.52	12.53 ± 0.08
+ AudioSet pretraining	13.18	35.30	48.61	22.80 ± 0.29
+ augmentations	14.50	37.24	51.04	24.27 ± 0.19
+ AudioCaps pretraining	14.34	38.12	52.04	24.57 ± 0.15

Table 3: Audio retrieval performance of the DCASE baseline and the custom system in four variants.

7.1. Baseline

The performance of our custom baseline system, which was trained with randomly initialized audio embedding model parameters and without augmentation, is given in Table 3. The resulting system's mAP is 5 pp. higher than the mAP of the DCASE baseline system [5] which we attribute to the more powerful text embedding model (we used BERT [8] instead of Word2Vec [24]).

7.2. AudioSet Pretraining

Next, we investigated the impact of using pretrained weights to initialize the audio embedding model. To that end, we retrained our baseline system but transferred the initial audio embedding model parameters from a CNN10 pretrained for tagging on AudioSet. The resulting model is approximately 10 pp. mAP better than the system that used randomly initialized weights for the audio encoder. This confirms that using pretrained audio embedding models is an effective strategy to alleviate the data scarcity problem.

7.3. Augmentations

We build upon the system that uses the AudioSet pretrained embedding model and perform SMBO to find a good hyperparameter configuration. The resulting best hyperparameters on the validation set and the performance on the test set are given in Tables 4 and 3, respectively. The best configuration found suggests that the parameter which controls the frequency of EDA is superfluous as the best value is very close to one. Synonym replacement appears beneficial, and future experiments should search for the optimal value for this parameter in a larger range. The probability of swapping and inserting random words is close to zero, suggesting that these two transformations are less beneficial or even detrimental. All in all, we observed an absolute improvement of approximately 1.5 pp. mAP when training with text and audio augmentations.

	Augmentation	Parameter	Range	best
Text	EDA	p_{EDA}	$[0, 1.0]$	.9936
		p_{syn}	$[0, 0.3]$	.2962
		p_{swp}	$[0, 0.3]$	.0085
		p_{ins}	$[0, 0.3]$	.0269
		p_{del}	$[0, 0.3]$	.1944
	Backtranslation	p_{bt}	$[0, 1.0]$	.1812
Audio	SpecAugment	n_{f}	$\{0, 1\}$	1
		w_{f}	$\{1, \ldots, 32\}$	4
		n_{t}	$\{0, \ldots, 8\}$	7
		w_{t}	$\{1, \ldots, 64\}$	58
	Audio Gain	g_{max}	$\{0, \ldots, 6\}$	3
	Freq-MixStyle	p_{MS}	$[0, 1.0]$	.1045
		α	$[0, 1.0]$	.8286

Table 4: Hyperparameter search space for the sequential model-based optimization, and the best configuration found.

7.4. AudioCaps Pretraining

We hypothesized that pretraining the retrieval system on additional audio-caption pairs could further improve audio-retrieval results; we, therefore, pretrained the system (with AudioSet pretraining and augmentations) on the $46K$ training examples in AudioCaps [25]. To this end, we used the same training procedure as described in Section 6.4 for pretraining and fine-tuning but decreased the initial learning rate for fine-tuning by a factor of 10. Table 3 gives the results. Pretraining on AudioCaps resulted only in a marginal improvement of 0.3pp mAP, which suggests that using AudioCaps for transfer learning in this naive way has no significant impact.

7.5. Ablation Study: Augmentations

Based on the previous results, we performed another ablation study to investigate the effect of the audio and text augmentations. To that end, we re-trained the system with AudioSet pretraining and augmentations twice: once without audio augmentations and once without text augmentations. The results are given in Table 5. Using all augmentations gave the best results. We observed a drop of 1.1 and 0.5 pp mAP without text and audio augmentations, respectively. This might indicate that the text augmentations have a larger impact than the audio augmentations, which might be caused by the large difference in trainable parameters between the sentence and audio embedding models.

To isolate the effect of each individual augmentation method, we further re-trained the system (with AudioSet pretraining and augmentations) in five variants, always leaving out one of the augmentation methods. The results are summarized in Table 5. The text augmentations have the largest impact, which is in line with the previous results: Leaving out EDA and BT reduced the mAP by 1.0 and 0.7 pp., respectively. Eliminating SpecAugment and Freq-MixStyle reduced the performance by 0.7 and 0.6 pp., respectively. Gain augmentation seems to have the least impact: eliminating it reduced the mAP by only 0.2pp.

	R@1	R@5	R@10	mAP@10
SMBO	14.50	37.24	51.04	24.27 ± 0.19
no audio aug	13.88	36.94	51.06	23.74 ± 0.16
no text aug	13.12	35.77	49.25	22.91 ± 0.08
no SpeAugment	13.50	36.60	50.91	23.53 ± 0.20
no FreqMixStyle	13.61	36.91	50.69	23.62 ± 0.14
no Gain Augment	14.84	37.81	50.95	24.05 ± 0.26
no BT	13.61	36.38	50.00	23.43 ± 0.18
no EDA	13.33	37.02	49.94	23.27 ± 0.03

Table 5: Results of the ablation study on data augmentation.

8. CONCLUSION

This study set out to investigate transfer learning and data augmentation strategies to alleviate the data scarcity problem in natural-language-based audio retrieval. Our research has shown that using pretrained audio and text embedding models greatly increases the retrieval performance on ClothoV2. We enriched this already well-performing retrieval system with a range of augmentation methods and showed that augmenting both text and audio inputs significantly reduces overfitting. Finally, we further found that pretraining on AudioCaps only leads to non-significant improvements.

9. ACKNOWLEDGMENT

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

10. REFERENCES

[1] A. Oncescu, A. S. Koepke, J. F. Henriques, Z. Akata, and S. Albanie, "Audio retrieval with natural language queries," in *22nd Annual Conf. of the Int. Speech Communication Association, Interspeech*, 2021.

[2] H. Xie, O. Räsänen, K. Drossos, and T. Virtanen, "Unsupervised audio-caption aligning learns correspondences between individual sound events and textual phrases," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process., ICASSP*, 2022.

[3] B. Elizalde, S. Deshmukh, M. A. Ismail, and H. Wang, "CLAP: learning audio concepts from natural language supervision," *CoRR*, vol. abs/2206.04769, 2022.

[4] 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," in *Proc. of the 38nd Int. Conf. on Machine Learning, ICML*, 2021.

[5] X. Huang, S. Lipping, and T. n. Virtanen, "DCASE 2022 Challenge Task 6b: Language-Based Audio Retrieval," *CoRR*, vol. abs/2206.06108, 2022.

[6] D. Turnbull, L. Barrington, D. A. Torres, and G. R. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Trans. Speech Audio Process.*, 2008.

[7] 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.

[8] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proc. of the North American Ch. of the Ass. for Computational Linguistics: Human Language Technologies, NAACL-HLT*, 2019.

[9] 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.

[10] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. of the 37nd Int. Conf. on Machine Learning, ICML*, 2020.

[11] K. Zhou, Y. Yang, Y. Qiao, and T. Xiang, "Domain generalization with mixstyle," in *9th Int. Conf. on Learning Representations, ICLR*, 2021.

[12] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics, ACL*, 2016.

[13] J. W. Wei and K. Zou, "EDA: easy data augmentation techniques for boosting performance on text classification tasks," in *Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int. Joint Conf. on Natural Language Processing, EMNLP-IJCNLP*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., 2019.

[14] G. A. Miller, "Wordnet: A lexical database for english," *Commun. ACM*, 1995.

[15] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: an Audio Captioning Dataset," in *Proc. IEEE Int. Conf. Acoustic., Speech and Signal Process., ICASSP*, 2020.

[16] 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.

[17] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. abs/1609.08144, 2016.

[18] 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. IEEE Int. Conf. Acoustic., Speech and Signal Process., ICASSP*, 2017.

[19] Y. Zhu, R. Kiros, R. S. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *IEEE Int. Conf. on Computer Vision, ICCV*, 2015.

[20] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," in *5th Int. Conf. on Learning Representations, ICLR*, 2017.

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

[22] 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.

[23] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Annual Conf. on Neural Information Processing Systems, NEURIPS*, 2011.

[24] 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.

[25] 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.