Multitask learning in Audio Captioning: a sentence embedding regression loss acts as a regularizer

Etienne Labbé, Julien Pinquier, Thomas Pellegrini

Multitask learning in Audio Captioning: a sentence embedding regression loss acts as a regularizer

Etienne Labbé
IRIT, Université de Toulouse
CNRS, UT3, Toulouse, France
etienne.labbe@irit.fr

Julien Pinquier
IRIT, Université de Toulouse
CNRS, UT3, Toulouse, France
julien.pinquier@irit.fr

Thomas Pellegrini
IRIT, Université de Toulouse
CNRS, UT3, Toulouse, France
thomas.pellegrini@irit.fr

***Abstract*— In this work, we propose to study the performance of a model trained with a sentence embedding regression loss component for the Automated Audio Captioning task. This task aims to build systems that can describe audio content with a single sentence written in natural language. Most systems are trained with the standard Cross-Entropy loss, which does not take into account the semantic closeness of the sentence. We found that adding a sentence embedding loss term reduces overfitting, but also increased SPIDEr from 0.397 to 0.418 in our first setting on the AudioCaps corpus. When we increased the weight decay value, we found our model to be much closer to the current state-of-the-art scores, with a SPIDEr score up to 0.444 compared to a 0.475 score. Moreover, this model uses eight times less trainable parameters than the current state-of-the-art method Multi-TTA. In this training setting, the sentence embedding loss has no more impact on the model performance.**

***Index Terms*—sound event description, multitask learning, audio language task, overfitting, sentence embedding regression loss, semantic loss**

I. Introduction

In recent years, new machine learning systems have been significantly improved for text processing, generation, and understanding, leading to the use of natural language as a global interface between humans and machines. Free-form text can contain much more information than a predefined set of classes, which could improve the machine understanding of our world. In audio, most of the tasks are focused on classification and localization of sound events. Following this idea, the Automated Audio Captioning (AAC) task appeared in 2017 [1] and aims to create systems that generate a sentence written in natural language that describes an audio file. The audio can contain various sound events (human, natural, domestic, urban, music, effects...) of different lengths, recorded with different devices and in different scenes. The description can contain any kind of detail in the audio, with temporal or spatial relations between them (followed by, in the background...) or different characterizations (high-pitched, short, repetitive...). Since the descriptions are written by humans, we need to consider different words used to describe similar sounds (*Birds are calling / chirping / singing / tweeting*), different sentence structures (*A door that needs to be oiled / A door with squeaky hinges*), subjectivity (*Man speaks in a foreign language*), high-level descriptions (*A vulgar man speaks / Unintelligible conversation*), and vagueness (*Someone speaks* instead of *A man gives a speech over a reverberating microphone*).

In AAC, most approaches use deep learning models trained with the standard Cross-Entropy (CE) loss [2]. However, this loss tends to generate repetitive and generic content [3] and does not take into account synonyms, various sentences structures or the semantic closeness. Several studies introduced another criterion, the Self-Critical Sequence Training [4] (SCST) used in reinforcement learning to fine-tune the model directly on a metric instead of the loss. This technique relies on sampling the next word to generate a new sentence. If this sentence has a higher score than the original one, the model is rewarded and the outputs probabilities for this new sentence are encouraged. However, this technique leads to degenerated sentences [5], with repetitive n-grams without syntactical correctness.

Motivated by the limitations of CE and SCST, in this work, we attempted to add a Sentence Embedding Regression (SER) loss used in [6] to improve our model. We begin this paper by describing our baseline system and then explain how to add SER loss. We present related work in which we compare and then describe the detailed hyperparameters. Finally, we present the results and discuss the differences.

II. Baseline system description

We use an encoder-decoder architecture widely used in AAC systems, with an encoder pre-trained on AudioSet [7] to extract a strong representation of sounds events. More specifically, we used the `CNN14_DecisionLevel_Att` audio encoder from the Pre-trained Audio Neural Networks study (PANN) [8]. This architecture gives the best results on the classification of sound events in the audio captioning dataset part when compared to the other PANN architectures available. We have found that freezing weights does not decrease performance while significantly speeding up the training process. This encoder provides sequences of embeddings of dimension 31×2048 for ten-second long audio recordings. On top of that, we add a projection layer to get 256-dimensional embeddings to match the input dimension of the decoder d_{model}.

The decoder is a standard transformer decoder [9] with 6 layers, 4 attention heads per layer, a global embedding size d_{model} set to 256 and a global dropout probability of 0.2. We also used the GELU [10] activation layer in the decoder.

The decoder is trained using teacher forcing, *i.e.* gives the ground truth previous reference tokens to the model to predict

the next one. The baseline criterion is the standard CE loss over the whole sequence between the output probabilities and the reference token classes.

During inference, we used the beam search algorithm with a beam size set to 2 since higher value does not bring improvements. We conditioned the sentence generation to improve performance and overall caption quality: by limiting the prediction length to a minimum of 3 tokens and a maximum of 30 tokens and by forbidding the model to generate the same token twice, except for stop-word tokens predefined in the Natural Language ToolKit (NLTK) [11] package. These constraints reduce the number of invalid sentences and repetitions and give a slight improvement in the performance of our model.

III. Adding a sentence embedding regression loss

A. Sentence-BERT model

The Sentence-BERT [12] (SBERT) model is a transformer-based model which combines a BERT [13] model with a pooling and a projection layer to produce a single embedding of a fixed size of 768 values for a given sentence.

Available SBERT models have been trained on two text databases: the Stanford Natural Language Inference (SNLI) [14] and the Multi-Genre Natural Language Inference (MultiNLI) [15]. These datasets contain pairs of two sentences annotated with contradiction, entailment or neutral label. To learn sentence semantic, two SBERT embedding are fed to a classification layer, which must predict the label of the pair.

B. SER loss

To use the SBERT model to improve our model, we need to use the same token units to have the same sequence size. BERT uses WordPiece tokens [16] instead of words which form a vocabulary of 30522 different units in our experiments.

Fig. 1 resumes the whole procedure and layers used. During training phase, we use audio features and the ground truth previous tokens to generate the next token embeddings named $\hat{e}_t$. These embeddings are used in two different parts of the model. First, they are projected to logits using a classifier for the standard CE loss $\mathcal{L}_t$. The token embeddings are also projected from 256 to 768-dimensional embedding to match the SBERT embedding input shape. The resulting embedding $\hat{e}_s$ is used as input with the ground truth embedding for the SER loss component $\mathcal{L}_s$. In order to use the SBERT model to train our model, we need to remove the first layer of SBERT which maps tokens IDs to embedding vectors (named "Embed" in the figure), since this layer is not differentiable. At inference time, only the classifier branch is used.

We tried several regression criteria for the $\mathcal{L}_s$ loss: CosineEmbeddingLoss, L1Loss, MSELoss and SmoothL1Loss. The best one that we obtained is the SmoothL1Loss [17], a regression function which combines MSE and L1Loss, described in (1).

$$\mathcal{L}_s(\hat{e}_s, e_s) = \begin{cases} (\hat{e}_s - e_s)^2 \cdot \frac{1}{2\beta} & \text{if } |\hat{e}_s - e_s| < \beta \\ |\hat{e}_s - e_s| - \frac{\beta}{2} & \text{otherwise} \end{cases} \quad (1)$$

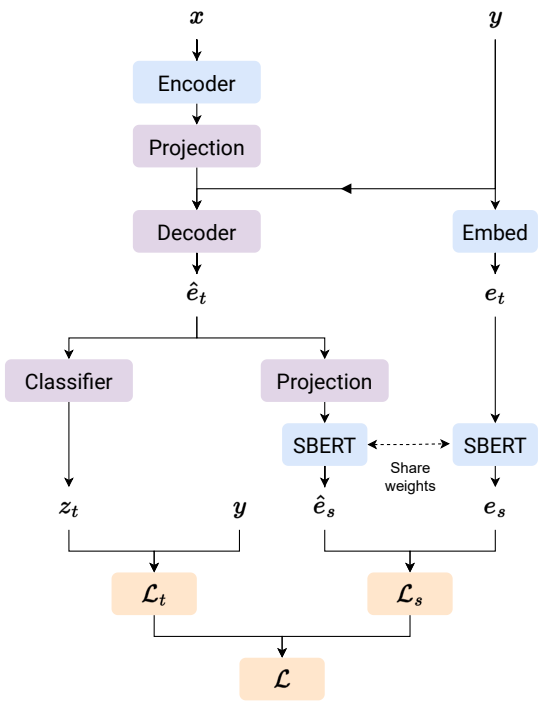


Fig. 1. Overview of our proposed training method. x denotes the input audio and y the corresponding reference. The blue boxes are the pre-trained frozen layers, the purple ones the trainable layers and the orange ones the loss functions. The SBERT block contains the SBERT model with its first embedding layer removed.

The β hyperparameter control whether MSE or L1Loss must be used. We kept the standard CE as our first component $\mathcal{L}_t$ to help the model to produce syntactically valid sentences. The final loss is given by (2) and sum $\mathcal{L}_t$ and $\mathcal{L}_s$, weighted by a coefficient λ:

$$\mathcal{L} = \mathcal{L}_t + \lambda \cdot \mathcal{L}_s \quad (2)$$

IV. Related work

The current state-of-the-art on AudioCaps [18] is a full transformer architecture named Audio Captioning Transformer (ACT) [19], pre-trained on AudioSet like PANN models. The system uses mixup [20] to improve generalization during training between audio waveform and spectrograms and concatenate the corresponding captions. For inference, Gaussian noise and SpecAugment [21] are used to produce several variants of the same sample and are given to the model. The intermediate representations of the same example are averaged to produce a better sentence.

In [22], the authors proposed to use a pre-trained transformer decoder named BART [23] to generate better sentences. Their first encoder (YAMNet) predicts the names of the predicted AudioSet classes to improve the audio representation. These classes names are the inputs of the BART embedding layer and are added to the PANN encoder audio embeddings. The decoder is the pretrained BART transformer decoder

part. We named their model BYP in the table for shortening BART+YAMNet+PANNs.

An approach similar to ours has been proposed in [6]. There are notable differences (the audio encoder, optimizer and hyperparameters for optimization and generation) which provide a stronger baseline. Unlike them, we train our model with only one phase (CE+SER losses) instead of two (CE then CE+SER losses).

V. Experimental setup

A. Dataset

We train and evaluate our models on the AudioCaps [24] dataset, which is the largest known audio-language dataset with human generated captions. The audio files are 10-second clips from AudioSet [7] and are extracted from YouTube videos. Since some of the original videos are removed or unavailable, our version of the dataset contains 46230 over 49838 files in training subset, 464 over 495 in validation subset and 912 over 975 files in testing subset. Each audio is described by one caption in the training subset and five captions in the validation and testing subsets.

B. Metrics

We focused only on the captioning metrics and decided to dismiss the translation metrics (BLEU, ROUGE, METEOR) which are mainly based on n-gram overlapping. CIDEr-D [25] computes the cosine similarity of the TF-IDF scores for common n-grams in candidates and references. SPICE [26] computes the F1-score of the graph edges representing the semantic propositions extracted from the sentences using a parser and grammar rules. SPIDEr [27] is the average of CIDEr-D and SPICE and mainly used to rank AAC systems. Since we are also studying a sentence similarity loss, we decided to add three model-based metrics from [28]: SBERT, FluErr and FENSE. The SBERT metric correspond to the cosine similarity of the sentence embedding extracted using a SBERT model. FluErr is the fluency error rate detected by a model trained to detect common errors made by captioning systems like incomplete sentence, repeated event, repeated adverb, missing conjunction and missing verb. The FENSE metric is the SBERT score for each sentence, unless an error in the FluErr metric is detected, then it will be divided by 10. Finally, the last metric "#Words" is the number of unique words used in the candidates in the whole subset.

C. Hyperparameters

Hyperparameters are crucial for training deep learning systems. We found that the model can obtain drastically different scores when trained using different sets of hyperparameters. We optimized our hyperparameters to maximize the FENSE score on the validation subset of AudioCaps. We train our model for a total of 100 epochs K and with a batch size of 512 samples on a single GPU. We used the AdamW optimizer [29] with a weight decay (wd) of 10^{-6} in the first experiments and set to 2 to limit overfitting in the second setting. The weight decay is not applied to the bias weights of the network. We also denote that the network does not converge when using the standard Adam [30] optimizer with a large wd. The initial learning rate lr_0 is set to $5 \cdot 10^{-4}$ at the beginning of the training, and the values of β_1 and β_2 are respectively set to 0.9 and 0.999. We used cosine scheduler decay updated at the end of each epoch k with the following rule: $\mathrm{lr}_k = \frac{1}{2}\big(1 + \cos(\frac{k\pi}{K})\big)\mathrm{lr}_0$. The captions are put in lowercase and all punctuation characters are erased. We clip the gradient by l2-norm to 10 to stabilize training and add label smoothing set to 0.1 to the CE loss component. To select our best model among epochs, we used the highest FENSE score instead of using the CE loss on the validation subset. Using FENSE allows you to choose a later training epoch than with the validation loss function, which gives better results.

For the sentence embedding regression method, we used the `paraphrase-TinyBERT-L6-v2` model from Hugging Face, since it is the one used in the metrics SBERT and FENSE. We also tried larger models (`all-mpnet-base-v2` and `all-mpnet-base-v1`), but it does not bring any improvements. The chosen model contains 67M parameters and its weights are frozen during training. We have set β to 1 in the $\mathcal{L}_s$ function. We tried several values for λ (1, 10, 100, 1000, and 10000), and found that 100 is the best parameter for sentence embedding regression. Higher values decrease performance, while lower values have no impact on multitasking compared to the baseline.

VI. Results

We reported the scores in Table I of our baseline method using a word tokenizer, the method using the SBERT tokenizer (baseline+SBERT tokens) and with the SER loss method (baseline+SBERT tokens+SER loss). All of our scores are averaged over 5 seeds. We also added the other SER method scores named CNN10-trans, the current state-of-the-art scores (Multi-TTA and BYP) and the cross-referencing scores. Cross-referencing is performed by excluding one of the five captions for each audio file and using it as a candidate sentence, while the other four remain the ground truth references. This process is repeated five times to compute an average human agreement score, which we call "Human".

A. Discussion

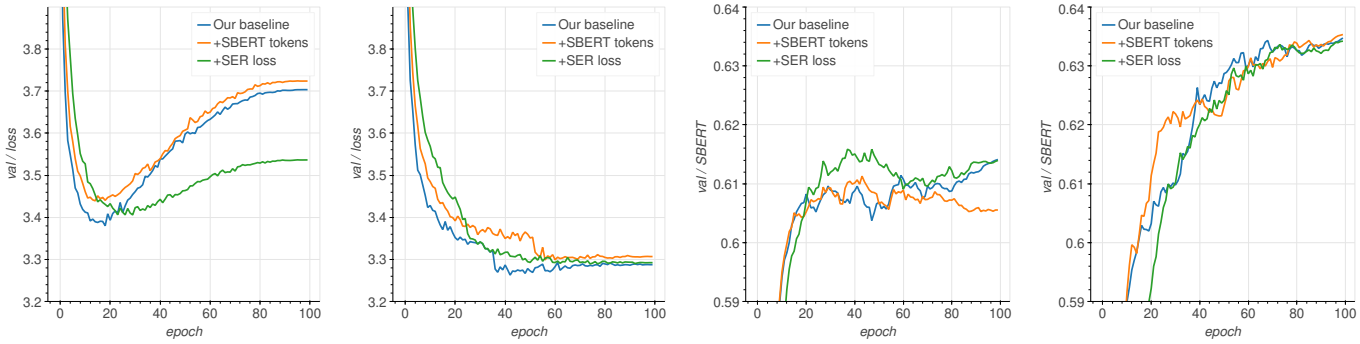
Using a small wd value, the SER loss shows a slight improvement in FENSE and SPIDEr scores respectively increased from 0.595 to 0.607 and from 0.397 to 0.418. Moreover, when we introduce a large wd value to prevent overfitting, we can see a large improvement in FENSE and SPIDEr scores from 0.595 to 0.619 and from 0.397 to 0.445 respectively with our baseline. Nevertheless, even if the SER loss also benefits from the use of a large wd value, the resulting scores became very close to the new baseline which also uses this wd value and the regularization effect given by the SER loss seems no longer significant.

In Fig. 2, we can see that the increase in the validation learning curve is reduced by the SER loss with a small wd, which means that it limits the overfitting of our model and

TABLE I

AAC RESULTS ON AUDIOCAPS **TESTING** SUBSET WITH CAPTIONING METRICS. THE ARROW ↑ INDICATES THAT A HIGHER VALUE IN THE COLUMN IS BETTER, WHILE ↓ INDICATES THAT A LOWER VALUE IS BETTER. OPTIM. AND WD. STAND FOR OPTIMIZER AND WEIGHT DECAY, RESPECTIVELY.

Method	**Optim.**	**Wd.**	**CIDEr-D** ↑	**SPICE** ↑	**SPIDEr** ↑	**FENSE** ↑	**SBERT** ↑	**FluErr** ↓	**#Words** ↑	**Trainable params**	**Frozen params**
Human	N/A	N/A	.901	.217	.559	.680	.682	.005	952.2	0	0
Multi-TTA [18]	AdamW	10^{-6}	**.769**	**.181**	**.475**	N/A	N/A	N/A	N/A	108M	0
BYP [22]		N/A	.753	.176	.465					408M	
CNN10-trans [6]	Adam	N/A	.573	.158	.365	N/A	.545	N/A	N/A	14M	0
+SER cosine loss			.573	.166	.370		.555				N/A
Our baseline	AdamW	10^{-6}	.628	.165	.397	.595	.601	.034	**485.8**	12.4M	79.7M
+SBERT tokens			.659	.167	.413	.602	.607	.034	445.0	25.7M	79.7M
+SER loss			.665	.170	.418	.607	.614	.027	465.6	25.9M	146.7M
Our baseline	AdamW	2	.712	**.176**	.444	.619	**.621**	.004	387.2	12.4M	79.7M
+SBERT tokens			**.715**	.175	**.445**	**.620**	**.621**	**.002**	390.2	25.7M	79.7M
+SER loss			**.715**	.172	.443	.619	.620	**.002**	348.8	25.9M	146.7M



Loss with wd set to 10^{-6}. Loss with wd set to 2. SBERT with wd set to 10^{-6}. SBERT with wd set to 2.

Fig. 2. CE losses and SBERT cosine similarities over epochs on validation.

explains the gain obtained in the table. The validation losses and SBERT cosine similarities in the figures show that the regularization with a large wd works very well on the model.

The increase in the number of trainable parameters between our baseline and our baseline+SBERT tokens (12M to 25.7M) comes from the increase in vocabulary (4724 to 30522) which drastically grows the number of parameters in the classifier and input embedding layer of the decoder. The method using SBERT tokens and large wd value has become our new best model according to SPIDEr and FENSE, although it is closely followed by our methods using the same decay. Our methods are also very close to the Multi-TTA method which obtain 0.475 compared to our best SPIDEr scores of 0.445 and 0.444, despite the fact that we have eight and four times fewer trainable parameters for our baseline and baseline+SBERT tokens, respectively.

B. Qualitative analysis

Tables II and III show several examples of the sentences generated by our model over different training procedures. The baseline system using small wd value seems to try to use more synonyms but fail more often to provide a good description, like in II. When using large wd value, the system uses even less words and more generic sentence structures, despite being more accurate. In this case, we also denote that the vocabulary size used in the testing subset decreased from an average of 485.8 words to 387.2 in our baselines systems using small and large wd value, respectively. The reduction is even more important when we add the SER loss, with only 348.8 words used.

TABLE II

CAPTIONS FOR AN AUDIOCAPS TESTING FILE (ID: "ARFFw0e_jig") USING THE WD VALUE 10^{-6} FOR DIFFERENT METHODS.

Candidates	**Method**	**SPIDEr**	**FENSE**
a person belching	Our baseline	.070	.208
a person burps loudly	SBERT tokens	.135	.291
a person burps loudly several times	SER loss	**.252**	**.398**
References			
loud burping and screaming			
loud burping repeating			
a loud distorted belch followed by a series of burping			
several distorted belches followed by non-distorted burps			
a series of distorted burps followed by non-distorted burps			

VII. CONCLUSIONS

In this work, we studied the addition of a sentence embedding regression loss component to improve an Automated Audio Captioning system. We searched for the most optimal configuration for our baseline by optimizing hyperparameters, conditioning generation and using a stronger pre-trained encoder. We discovered that the SER loss component seems to limit overfitting for AAC systems, but it does not bring

TABLE III
CAPTIONS FOR AN AUDIOCAPS TESTING FILE (ID: "JZloTOdIY_c") WITH DIFFERENT WEIGHT DECAYS OF OUR BASELINE.

Candidates	**wd**	**SPIDEr**	**FENSE**
a horse neighs and breathes heavily	10^{-6}	.284	**.658**
a horse is trotting	2	**.337**	.452
References			
horses growl and clop hooves			
a horse neighs followed by horse trotting and snorting			
horses neighing and snorting while trotting on grass			
horses neighing then snorting and trotting on a dirt surface			
horses neighing and stomping on the ground			

improvement anymore when combined with a stronger regularization method like a large weight decay value. We also noticed that even if the main metrics (SPIDEr, FENSE) are improved by the regularization methods, they do not take into account the diversity of the words used. The diversity of words used could be taken into account in future captioning metrics, or directly by the model during learning like in [31].

ACKNOWLEDGMENT

This work was partially supported by the Agence Nationale de la Recherche LUDAU (Lightly-supervised and Unsupervised Discovery of Audio Units using Deep Learning) project (ANR-18-CE23-0005-01), and was granted access to the HPC resources of IDRIS under the allocation 2022-AD011013739 made by GENCI.

REFERENCES

[1] K. Drossos, S. Adavanne, and T. Virtanen, "Automated audio captioning with recurrent neural networks," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2017, pp. 374–378.

[2] X. Mei, X. Liu, M. D. Plumbley, and W. Wang, "Automated audio captioning: an overview of recent progress and new challenges," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2022, no. 1, p. 26, Oct 2022.

[3] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," 2019.

[4] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, "Self-critical sequence training for image captioning," 2016.

[5] X. Mei, Q. Huang, X. Liu, G. Chen, J. Wu, Y. Wu, J. Zhao, S. Li, T. Ko, H. L. Tang, X. Shao, M. D. Plumbley, and W. Wang, "An encoder-decoder based audio captioning system with transfer and reinforcement learning," 2021.

[6] R. Mahfuz, Y. Guo, and E. Visser, "Improving audio captioning using semantic similarity metrics," 2022.

[7] 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 International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. New Orleans, LA: IEEE, Mar. 2017, pp. 776–780.

[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," *arXiv:1912.10211 [cs, eess]*, Aug. 2020, arXiv: 1912.10211.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[10] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," 2016.

[11] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[12] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," 2019.

[13] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018.

[14] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 632–642.

[15] A. Williams, N. Nangia, and S. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1112–1122.

[16] 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," 2016.

[17] R. B. Girshick, "Fast R-CNN," *CoRR*, vol. abs/1504.08083, 2015.

[18] E. Kim, J. Kim, Y. Oh, K. Kim, M. Park, J. Sim, J. Lee, and K. Lee, "Improving audio-language learning with mixgen and multi-level test-time augmentation," 2022.

[19] X. Mei, X. Liu, Q. Huang, M. D. Plumbley, and W. Wang, "Audio captioning transformer," 2021.

[20] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," 2018.

[21] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[22] F. Gontier, R. Serizel, and C. Cerisara, "Automated audio captioning by fine-tuning bart with audioset tags," in *DCASE 2021 - 6th Workshop on Detection and Classification of Acoustic Scenes and Events*, Virtual, Spain, Nov. 2021.

[23] 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," 2019.

[24] C. D. Kim, B. Kim, H. Lee, and G. Kim, "AudioCaps: Generating captions for audios in the wild," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 119–132.

[25] R. Vedantam, C. L. Zitnick, and D. Parikh, "CIDEr: Consensus-based Image Description Evaluation," *arXiv:1411.5726 [cs]*, Jun. 2015, arXiv: 1411.5726.

[26] P. Anderson, B. Fernando, M. Johnson, and S. Gould, "SPICE: Semantic Propositional Image Caption Evaluation," *arXiv:1607.08822 [cs]*, Jul. 2016, arXiv: 1607.08822.

[27] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, "Improved Image Captioning via Policy Gradient optimization of SPIDEr," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 873–881, Oct. 2017, arXiv: 1612.00370.

[28] Z. Zhou, Z. Zhang, X. Xu, Z. Xie, M. Wu, and K. Q. Zhu, "Can Audio Captions Be Evaluated with Image Caption Metrics?" Jan. 2022, number: arXiv:2110.04684 arXiv:2110.04684 [cs, eess].

[29] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[30] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980 [cs]*, Jan. 2017, arXiv: 1412.6980.

[31] S. Welleck, I. Kulikov, S. Roller, E. Dinan, K. Cho, and J. Weston, "Neural text generation with unlikelihood training," 2019.